Numeričke metode

Aleksandar Maksimović
IRB

Sadržaj

- Uvod
- Greške u numeričkom računanju
- Uvod u F90 (ili F77/C++)
- Sustavi linearnih jednadžbi
  - Gausova i LU metoda
- Interpolacija
- Numerička integracija
- Rješavanje nelinearnih jednadžbi

sadržaj

- iterativne metode
- Obične diferencijalne jednadžbe
  - Runge Kutta
  - Euler
  - sustavi diferencijalnih jednadžbi

Uvodni sadržaj

- Tipovi grešaka
- aritmetika s pomičnom točkom (floating-point)
- primjeri pogrešaka
- kompajleri
- programiranje u f77/f90/C/C++
- gnu make
- fortran 90

Greške u numeričkom računu

- Tipovi grešaka:
  - Greške zbog aproksimacija
    - Greške modela (zanemarivanje otpora zraka)
    - Greške diskretizacije (truncation error), zamjena integrala konačnom sumom, zamjena tangente sekantom i slično.
    - Greške u polaznim podacima
  - Greške zaokruživanja, (roundoff errors) konačna preciznost u računanju dovodi do zaokruživanja vrijednosti.
    - greške nastaju u računalima, jer ona koriste konačnu aritmetiku ili preciznije binarnu aritmetiku s pomičnom točkom, kod koje je unaprijed rezerviran određen broj binarnih mjesta za eksponent i za mantisu

Aritmetika s pomičnom točkom

Na kalkulatorima se često može izabrati tzv. "znanstvena notacija" brojeva, koja npr. broj –27.77 prikazuje kao $-2.777 \cdot 10^1$ pri čemu je – predznak broja (ili mantise), . je decimalna točka, 2.777 je mantisa, koja se još zove signifikantni ili razlomljeni dio broja, 10 je baza, a 1 je eksponent.

$$\begin{aligned} x &= -(2 \cdot 10^1 + 7 \cdot 10^0 + 7 \cdot 10^{-1} + 7 \cdot 10^{-2}) \\ &= -(2 \cdot 10^0 + 7 \cdot 10^{-1} + 7 \cdot 10^{-2} + 7 \cdot 10^{-3}) \cdot 10^1, \end{aligned}$$

binarna reprezentacija

$$\begin{aligned} y &= (11.1011)_2 = 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} \\ &= (1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} + 1 \cdot 2^{-5}) \cdot 2^1 \\ &= (1.11011)_2 \cdot 2^1. \end{aligned}$$

reprezentacija realnih brojeva

U računalu se realni brojevi reprezentiraju pomoću binarne reprezentacije u "znanstvenoj notaciji"

$$x = \pm m \times 2^e, \quad \text{gdje je} \quad 1 \leq m < 2.$$

Stoga je

$$m = (b_0.b_1b_2b_3\ldots b_{p-1})_2 \quad \text{pri čemu je} \quad b_0 = 1.$$

broj kao suma cijelog i razlomljenog dijela

$$\begin{aligned} x &= (a_k a_{k-1}\cdots a_1 a_0.b_1 b_2\cdots b_{l-1}b_l)_2 = x_c + x_r, \\ x_c &= (a_k a_{k-1}\cdots a_1 a_0)_2 \\ &= a_k\cdot 2^k + a_{k-1}\cdot 2^{k-1} + \cdots + a_1\cdot 2^1 + a_0\cdot 2^0, \\ x_r &= (.b_1 b_2\cdots b_{l-1}b_l)_2 \\ &= b_1\cdot 2^{-1} + b_2\cdot 2^{-2} + \cdots + b_{l-1}\cdot 2^{-(l-1)} + b_l\cdot 2^{-l}. \end{aligned}$$

pretvaranje realnog broja

algoritam za računanje binarnog prikaza realnog broja
u pseudokodu i dio algoritma implementiranog u fortranu

```
y := xc;
k := −1;
repeat
k := k + 1;
a(k) := mod(y, 2);
y := div(y, 2);
until y = 0;
```

```
z := xr;
l := 0;
repeat
l := l + 1;
zz := 2      z;
b(k) := int(zz);
z := zz − b(k);
until z = 0 or l > lmax
```

```
      aint = Int(a)
      adec = a-aint

Do i=1,50
      v(i)    = Mod(aint,two)
      nb(i-1) = Nint(v(i))
      aint    = 0.5D0*(aint-v(i))
      If(aint.le.0.00001) Go to 97
    End Do
97 Continue
```

pretvaranje realnog broja

pretvaranje broja 111.1 u binarni oblik

111	55	27	13	6	3	1	0
1	1	1	1	0	1	1	.

$111 = (110111)_2.$ krivo napisano (1101111)

0.1	0.2	0.4	0.8	1.6	1.2	0.4	0.8	1.6	1.2
.	0	0	0	1	1	0	0	1	1

$$0.1 = (0.0001100110011001100\ldots)_2 = (0.0\dot{0}01\dot{1})_2,$$

$$(111.1)_{10} = (1101111.0001100110011001100\ldots)_2 = (1101111.0\dot{0}01\dot{1})_2.$$

Primjeri grešaka

- Promašaj rakete patriot

U Zaljevskom ratu, 25. veljače 1991. godine, Patriot rakete iznad Dhahrana u Saudijskoj Arabiji nisu uspjele pronaći i oboriti iračku Scud raketu. Projektil je (pukim slučajem) pao na američku vojnu bazu usmrtivši 28 i ranivši stotinjak ljudi.

Izvještaj o katastrofi godinu dana poslije, rasvijetlio je okolnosti nesreće. U računalu koje je upravljalo Patriot raketama, vrijeme se brojilo u desetinkama sekunde proteklim od trenutka kad je računalo upaljeno. Kad desetinku prikažemo u binarnom prikazu, dobivamo

$$0.1_{10} = (0.0\dot{0}011\dot{1})_2.$$

Realne brojeve u tom računalu prikazivali su korištenjem nenormalizirane mantise duljine 23 bita. Spremanjem 0.1 u registar Patriot računala, napravljena je (apsolutna) greška približno jednaka $9.5 \cdot 10^{-8}$.

Primjeri grešaka

Zbog stalne opasnosti od napada Scud raketama, računalo je bilo u pogonu 100 *sati, što je* $100 \cdot 60 \cdot 60 \cdot 10$ *desetinki sekunde. Ukupna greška nastala greškom zaokruživanja je*

$$100 \cdot 60 \cdot 60 \cdot 10 \cdot 9.5 \cdot 10^{-8} = 0.34\,\mathrm{s}.$$

Ako je poznato da Scud putuje brzinom $1676\,\mathrm{m/s}$ *na predviđenoj visini susreta, onda su ga rakete Patriot pokušale naći više od pola kilometra daleko od njegovog stvarnog položaja.*

Tako je sistemski sat mjerio vrijeme u desetinkama sekunde, ali se vrijeme spremalo kao cijeli broj desetinki proteklih od trenutka kad je računalo upaljeno. Za sve ostale proračune, vrijeme se računa tako da se pomnoži broj desetinki n s osnovnom jedinicom $t_0 = 0.1\,\mathrm{s}$ *u kvazi-cjelobrojnoj aritmetici, a zatim pretvori u pravi 24-bitni floating point broj.*

Primjeri grešaka

- Eksplozija Ariane 5 str28

Raketa Ariane 5 lansirana 4. lipnja 1995. godine iz Kouroua (Francuska Gvajana) nosila je u putanju oko Zemlje komunikacijske satelite vrijedne oko 500 milijuna USD. Samo 37 sekundi nakon lansiranja izvršila je samouništenje.

Dva tjedna kasnije, stručnjaci su objasnili događaj. Kontrolna varijabla (koja je služila samo informacije radi) u programu vođenja rakete mjerila je horizontalnu brzinu rakete. Greška je nastupila kad je program pokušao pretvoriti preveliki 64-bitni realni broj u 16-bitni cijeli broj. Računalo je javilo grešku, što je izazvalo samouništenje. Zanimljivo je da je taj isti program bio korišten u prijašnjoj sporijoj verziji Ariane 4, pa do katastrofe nije došlo.

Primjeri grešaka

- Potonuće naftne platforme Sleipner

Naftna platforma Sleipner A potonula je prilikom sidrenja, 23. kolovoza 1991. u blizini Stavangera. Baza platforme su 24 betonske ćelije, od kojih su 4 produljene u šuplje stupove na kojima leži paluba. Prilikom uronjavanja baze došlo je do pucanja. Rušenje je izazvalo potres jačine 3.0 stupnja po Richterovoj ljestvici i štetu od 700 milijuna USD.

Greška je nastala u projektiranju, primjenom standardnog paketa programa, kad je upotrijebljena metoda konačnih elemenata s nedovoljnom točnošću (netko nije provjerio rezultate programa). Proračun je dao naprezanja 47% manja od stvarnih. Nakon detaljne analize s točnijim konačnim elementima, izračunato je da su ćelije morale popustiti na dubini od 62 metra. Stvarna dubina pucanja bila je 65 metara!

Izabran je pogrešan predsjednik

Možda je najbizarniji primjer da greška zaokruživanja može poremetiti izbore za predsjednika SAD. U američkom sustavu izbora predsjednika, svaka od saveznih država ima određen broj predstavnika (ljudi) u tijelu koje se zove Electoral College i koje formalno bira predsjednika. Broj predstavnika svake pojedine države u tom tijelu proporcionalan je broju stanovnika te države u odnosu na ukupan broj stanovnika. Pretpostavimo da u Electoral College-u ima a predstavnika, populacija SAD je p stanovnika, a država i ima p_i stanovnika. Broj predstavnika države i u Electoral College-u trebao bi biti

$$a_i = \frac{p_i}{p} \cdot a.$$

Ali, predstavnici su ljudi, pa bi a_i morao biti cijeli broj. Zbog toga se a_i mora zaokružiti na cijeli broj $\hat{a}_i$ po nekom pravilu. Naravno, na kraju mora biti $\sum_i \hat{a}_i = a$. Razumno i prirodno pravilo je:

- *$\hat{a}_i$ mora biti jedan od dva cijela broja koji su najbliži a_i (tzv. "uvjet kvote").*

Primjeri grešaka

- Izabran je pogrešan predsjednik

Naime, pravilno zaokruživanje (kao kod prikaza brojeva) je možda najpravednije, ali ne mora dati $\sum_i \hat{a}_i = a$, pa se mora upotrijebiti slabije pravilo.

Međutim, broj stanovnika p_i se vremenom mijenja (a time i p). Isto tako, ukupni broj predstavnika a u tijelu se može promijeniti od jednih do drugih izbora. Zbog toga se dodaju još dva prirodna "politička" pravila:

- *Ako se poveća ukupan broj predstavnika a, a svi ostali podaci se ne promijene, $\hat{a}_i$ ne smije opasti (tzv. "monotonost predstavničkog tijela").*
- *Ako je broj stanovnika države p_i porastao, a ostali podaci su nepromijenjeni, $\hat{a}_i$ ne smije opasti (tzv. "monotonost populacije").*

Svrha je jasna, jer ljudi vole uspoređivati prošle i nove "kvote". Nažalost, ne postoji metoda za određivanje broja predstavnika koja bi zadovoljavala sva tri kriterija.

Primjeri grešaka

- Izabran je pogrešan predsjednik

U američkoj povijesti zaista postoji slučaj da je izabran "pogrešan" predsjednik. Samuel J. Tilden izgubio je izbore 1876. godine u korist Rutherforda B. Hayesa, samo zbog načina dodjele elektorskih glasova u toj prilici. Da stvar bude još zanimljivija, ta metoda dodjele glasova nije bila ona koju je propisivao zakon iz tog vremena.

Kompajleri

- Linux
  - ovisno o distribuciji
    - gnu kompajleri za C,f77 kod starijih distribucija
    - dominis.phy.hr ima gcc i g++ kompajlere za c i c++, te f77 za fortran.
    - fedora core 4, gcc, g++, ver. 4.02. Fortran kompajler gfortran
- Intel (linux)
  - besplatni kompajleri, registracija je obavezna

Kompajleri

- odlična dokumentacija, laka instalacija
- ftp site: ftp://download.intel.com/software/products/compilers/downloads/
- Free software for linux open source developers: registracija http://www.intel.com/cd/software/products/asmo-na/eng/219771.htm
- Fortran kompajler je ifort, icc je kompajler za C i C++
- debuger je idb, a najnovija verzija 9.0.x ima i eclipse kao grafičko sučelje za programiranje.
- integrira se sa gnu kompajlerima

Kompajleri osnovne opcije

- Gnu i Intel kompajleri imaju iste osnovne opcije
  - gcc file.c napravi a.out program za izvršavanje
  - gcc -o exefile file.c, exefile je rezultat
  - gcc -c file.c, napravi object file file.o
  - gcc -o exefile file.o, link object file i napravi exefile
  - gcc -I /includefiles/ -I/josinclude -L/libraries -o exef file.c -lml, pri kompajliranju file.c uključi header fileove iz direktorija navedenih nakon -I i linka s libml.a datotekom.

Kompajleri osnovne opcije

- Gnu i Intel kompajleri imaju iste osnovne opcije
  - -c - compile
  - -o - konačni program
  - -L - mjesto gdje se nalaze library (.a na linuxu, .so dynamicki)
  - -I - mjesto dodatnih include fileova
  - -O[X] - optimizacija gdje je X=2,3 itd. ovisno o kompajleru
  - -E - preprocesiranje
  - -g - uključi informacije za debug u object file
  - -- help pomoć za gnu kompajlere, info program sadrži dodatne informacije o gnu kompajlerima i datotekama

Kompajleri osnovne opcije

- -help pomoć za Intel kompajlere, kompletan opis opcija i jezika u html i pdf formatu
- DEBUG
  - GNU, gdb, linijski editor
    - ddd, grafičko sučelje za gdb
    - eclipse kod novijih distribucija (IDE)
  - INTEL
    - idb, linijski editor, eclipse (IDE)

primjeri programa f77

- hello program: hello.f

```
      program fortranprint
c komentar
      write(*,*)  "Hello World!"
c prolazi i print *, "string"
      end program
```

- pretvara decimalni broj u binarni oblik: binary.f

EXE file: g77 -o hellof77 hello.f

primjeri programa f90

- hello program: hello.f90

```
program fortranprint
c komentar
      print *,  "Hello World!"
      end program
```

- površina trokuta: triangle.f90

EXE file: gfortran -o hellof90 hello.f90

ili neki drugi kompajler (ifort) ili f90 od NAG grupe.

primjeri programa C

- hello.c

```
#include <stdio.h>
int main()
{  printf("Hello World!\n");
return 0; }
```

EXE file: gcc -o helloC hello.c

primjeri programa C++

- helo.cc

```
#include <iostream>
using namespace std;

int main()
{ cout << "Hello, World\n";
return 0; }
```

EXE file: g++-3.3 -o helloCC hello.cc

make i makefile

- Makefile se može koristiti za specificiranje opcija kod kompajliranja
  - sve naredbe s komandne linije se mogu koristiti (shell)
  - ima jednostavnu strukturu
  - VAR= lista , $(VAR) - ekspandira listu

```
all: EXEfile
CC=gcc
SRC=file.c
```

```
file.o: file.c
	$(CC) -c file.c
EXEfile: file.o
	$(CC) -o file.EXE fileo.o -L/libs -lm
```

Top 10 algoritama

- Dongarra & Sullivan sastavili su listu najvažnijih algoritama

1. **1946:** The Monte Carlo method for modeling probabilistic phenomena.
2. **1947:** The Simplex method for linear optimization problems.
3. **1950:** The Krylov subspace iteration method for fast linear solvers and eigensolvers.
4. **1951:** The Householder matrix decomposition to express a matrix as a product of simpler matrices.
5. **1957:** The FORTRAN compiler that liberated scientists and engineers from programming in assembly.
6. **1959-1961:** The QR algorithm to compute many eigenvalues.
7. **1962:** The Quicksort algorithm to put things in numerical or alphabetical order fast.

Top 10 algoritama

8. **1965:** The Fast Fourier Transform to reduce operation count in Fourier series representation.

9. **1977:** The Integer relation detection algorithm, which is useful for bifurcations and in quantum field theory.

10. **1987:** The Fast multipole algorithm for N-body problems.

Fortran kratka povijest

- 1950- *IBM Mathematical FORmula TRANslation System*

(prvi jezik “višeg stupnja”)

- do 1963 bilo je 40 različitih prevodilaca (compilers)
- 1972 prvi standard “FORTRAN 66”
- 1980 izašao “FORTRAN 77”
- FORTRAN 90 sadrži FORTRAN 77 kao podskup
- FORTRAN 95 (High Performance Fortran (HPF))

F77 nedostaci/karakteristike

- Prvih pet stupaca rezervirani;
- 6 stupac rezerviran za oznaku nastavka prethodnog reda;
- Komentari se inicijaliziraju slovom u prvom stupcu;
- Samo velika slova;
- Imena varijabli mogu imati do 6 znakova.
- Komentari moraju biti u posebnim redovima;
- Nema dinamičko spremanje (dimenzije polja su fiksne)
- Korisnik ne može definirati svoj tip varijable
- Ne podržava rekurziju

F90 nove osobine

- dinamičko alociranje memorije
- moduli
- pointeri
- slobodni format
- izvedene strukture
- CASE naredba
- IMPLICIT NONE
- kompatibilan sa f77

F95 nove osobine

- naredba FORALL
- inicijalizacija izvedenih struktura
- bolja kompatibilnost sa IEEE aritmetikom
- A CPU_TIME subroutine
- A function NULL to nullify a pointer
- automatska dealokacija alociranog niza iz bloka programa
- incijalizacija pointera
- itd...

f90

Fortran 90 podržava slobodan format:

- 132 znaka po redu;
- Proširen skup znakova;
- `&' znak nastavka reda;
- `!' početak komentara;
- `;' razdvajanje naredbi;
- significant (blanks).

f90 imena

- A, aAa, INCOME, Num1, N12O5, under_score ! O.K.
- Slijedeće deklaracije su nevažeće zbog nevažećih imena:
  - INTEGER :: 1A ! Ne počinje sa slovom
  - INTEGER :: A_name_made_up_of_more_than_31_letters !
  - INTEGER :: Depth:0 ! Sadrži nevažeći znak ":"
  - INTEGER :: A-3 ! Interpretira se kao A minus 3
  - CHARACTER(LEN=12) :: user_name! O.K.
  - CHARACTER(LEN=12) :: username ! O.K.

f90 deklaracija varijabli

```
CHARACTER(LEN=10) :: name
CHARACTER :: sex
CHARACTER(LEN=32) :: str
CHARACTER(LEN=10) :: name, spol*1, str*32
CHARACTER(LEN=10), DIMENSION(10,10) :: tom(10)*2, harry(10,10,10)*20
CHARACTER, POINTER :: P2ch

INTEGER :: NumBabiesBorn = 0 ! Zbroj novorođenčadi
REAL :: TimeElapsed = 0.0
LOGICAL :: NHS !
REAL, PARAMETER :: pi = 3.141592
REAL :: radius = 3.5
REAL :: circum = 2 * pi * radius
INTEGER :: a(1:4) = (/1,2,3,4/)

COMPLEX :: val ! x + iy
```

f90 implicitna deklaracija

- Ako se varijabla upotrebljava bez deklaracije onda je tip određen prvim slovom
  - I, J, K, L, M i N su tipa integer;
  - Sva druga slova su tipa real.
- To je opasno jer krivo utipkavanje stvara novu varijablu koje korisnik nije svjestan .
- IMPLICIT NONE !isključuje implicitnu deklaraciju

Literatura

- Online literatura:
  - Numerička matematika-osnovni udžbenik, PMF, projekt mzt.
  - A F90 Tutorial, 1993., (Z. Dodson, Univ. of New Mexico)
  - Harold W. Schranz (html):
    NUMERICAL METHODS: FORTRAN & MATHEMATICA
  - C Pozirkidis: C++ for Fortran Programmers, lecture notes
  - George W. Collins, II:
    Fundamental Numerical Methods and Data Analysis

Literatura

- Michael Metcalf: Fortran 90/95 Explained, Oxford University Press (2002), ISBN 0-19-850558-2
- L. F. Shampine, R. C. Allen, Jr., S. Pruess: FUNDAMENTALS OF NUMERICAL COMPUTING, John Wiley & Sons, Inc. (1997)
- S. D. Conte, Carl de Boore: ELEMENTARY NUMERICAL ANALYSIS, McGraw-Hill Book Company, (1980).

Zadaci za praktikum

(a) Skompajlirajte i uvjerite se da radi barem jedan od hello programa iz teksta

i. Promjenite poruku koju ispisuje na ekranu, npr. neka ispiše "Numeričke metode, Prezime Ime"

(b) Skompajlirajte i uvjerite se da radi program triangle

(c) Promjenite program triangle tako da računa površinu kruga

(d) Pošaljite mail na Aleksandar.Maksimovic@irb.hr koji za attachmente ima source ova dva programa i napišite koji bi programski jezik voljeli koristiti.

(e) Dodatni programi: prime (c,cc,f) da li je broj prost (prime) broj, Overflow.f dodavanje cijelog broja najvećem cjelom broju i IntegerDivide.cc primjer krivog pridruživanja (treba napisati output).

triangle.f

```
FUNCTION Area(x,y,z)
    REAL  x, y, z
    REAL theta, height
    theta = ACOS((x**2+y**2-z**2)/(2.0*x*y))
    height = x*SIN(theta)
    Area = 0.5*y*height
    END FUNCTION Area


    PROGRAM Triangle
    REAL  a, b, c
    PRINT *, 'Welcome, please enter the lengths of the 3 sides.'
    READ    *, a, b, c
    PRINT *, 'Triangle"s area: ', Area(a,b,c)
    END PROGRAM Triangle
```

triangle.f90

```
MODULE TriangleOperations
   IMPLICIT NONE
CONTAINS
  FUNCTION Area(x,y,z)
    REAL :: Area ! function type
    REAL, INTENT( IN ) :: x, y, z
    REAL :: theta, height
    theta = ACOS((x**2+y**2-z**2)/(2.0*x*y))
    height = x*SIN(theta); Area = 0.5*y*height
  END FUNCTION Area
END MODULE TriangleOperations
```

```
PROGRAM Triangle
use TriangleOperations

   IMPLICIT NONE
   REAL :: a, b, c
   PRINT *, 'Welcome, please enter the&
                &lengths of the 3 sides.'
   READ    *, a, b, c
   PRINT *, 'Triangle"s area: ', Area(a,b,c)
END PROGRAM Triangle
```

triangle.c

```
#include <stdio.h>
#include <math.h>
#define real float

real  Area(float x,float y,float z)
{
  real theta, height,povrsina ;
  theta = acos((x*x+y*y-z*z)/(2.0*x*y)) ;
  height = x*sin(theta);
  povrsina = 0.5*y*height ;
  return povrsina;
}
```

```
int main()
{  real  a, b, c;
  printf("Welcome, please enter the
lengths of the 3 sides.\n");
  scanf("%f,%f,%f" , &a, &b, &c);
  printf("Triangle"s area: %f",
Area(a,b,c));
  return 1;}
```

triangle.cc

```
#include <iostream>
#include <math.h>
using namespace std;
#define real float

int main()
{  real  a, b, c;
   cout << "Welcome, please enter the lengths of the 3 sides.\n" << endl;
   cin >> a >> b >> c;
   cout << "Triangle"s area: " << Area(a,b,c) << endl ;
   return 1; }

real  Area(float x,float y,float z)
{
  real theta, height,povrsina ;
  theta = acos((x*x+y*y-z*z)/(2.0*x*y)) ;
  height = x*sin(theta);
  povrsina = 0.5*y*height ;
  return povrsina; }
```