NUMERIČKE METODE I MATEMATIČKO MODELIRANJE

3. PREDAVANJE

NUMERIČKO DERIVIRANJE

- Numeričko deriviranje i integracija su od iznimne važnosti u računalnoj fizici i općenito u znanosti

- Ovdje će biti uvedene neke od metoda za numeričko deriviranje, vodeći računa o numeričkoj preciznosti i potencijalnim uzrocima greške

- Računanje prve i druge derivacije, uvođenje formalizma koji nalazi ključnu primjenu u numeričkom rješavanju običnih i parcijalnih diferencijalnih jednadžbi

- Većinu diferencijalnih jednadžbi u modernoj znanosti (npr. od klasične do kvantne fizike, primjenjene znanosti) *potrebno je rješavati numeričkim metodama*

NUMERIČKO DERIVIRANJE

- matematička definicija derivacije funkcije f(x):

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

ovdje je h je veličina koraka

- Primjenom *Taylorovog razvoja* za funkciju f(x) dobivamo

$$f(x+h) = f(x) + hf'(x) + \frac{h^2 f''(x)}{2} + \dots$$

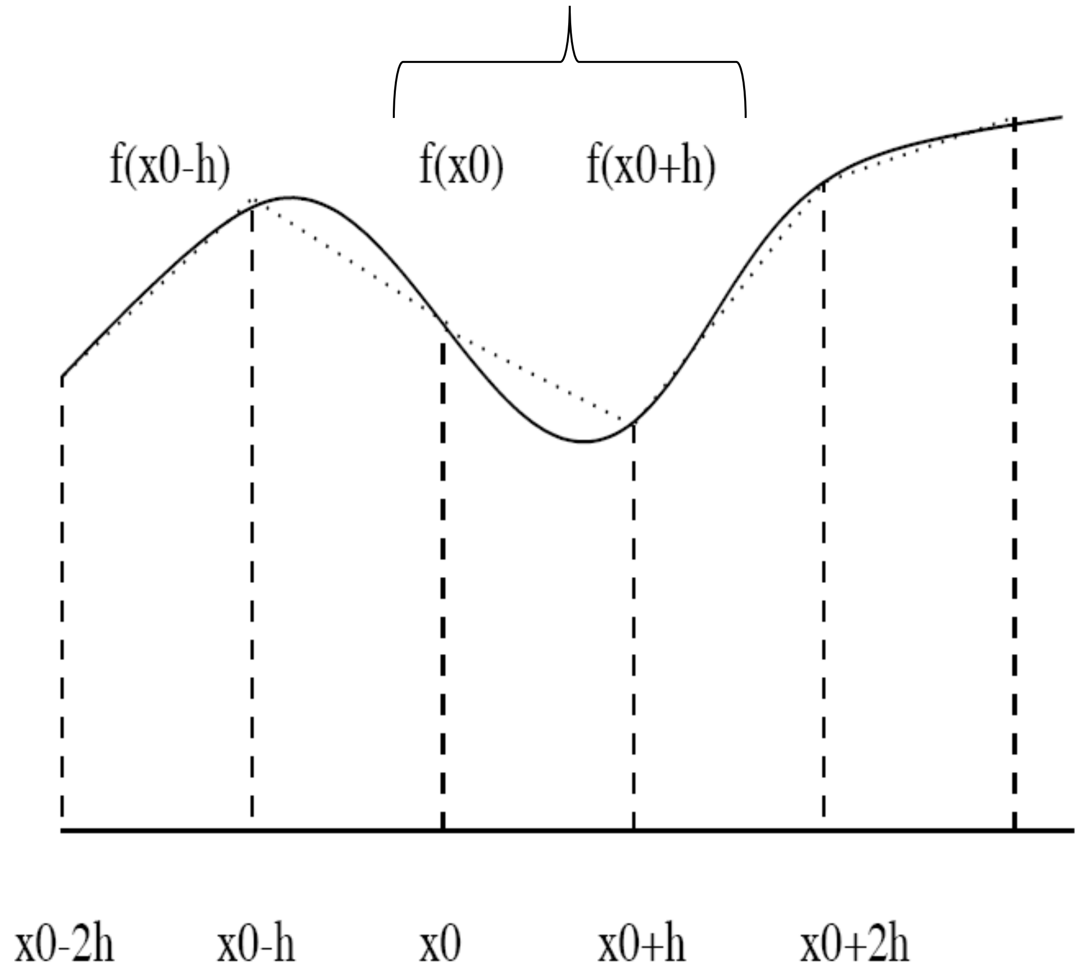
- derivaciju funkcije određene numerički označavamo $f'_c(x)$

$$f'_c(x) \approx \frac{f(x+h) - f(x)}{h} \approx f'(x) + \frac{hf''(x)}{2} + \dots$$

(Taylorov razvoj je uvršten u izraz za derivaciju)

NUMERIČKO DERIVIRANJE

- Pretpostavimo da u intervalu između x i x+h uzimamo dvije točke da prikažemo funkciju f (ravna linija)



PRVA DERIVACIJA POMOĆU DVIJE TOČKE

- U slučaju računanja derivacije pomoću dvije točke,

$$f_2'(x) = \frac{f(x+h) - f(x)}{h} + O(h)$$

- *O(h)* označava dominantnu grešku ovakvog proračuna
- Umjesto računanja prema naprijed, derivacija se može računati formulom prema nazad:

$$f_2'(x) = \frac{f(x) - f(x-h)}{h} + O(h)$$

- Ako je druga derivacija funkcije blizu nuli, ova jednostavna formula se može koristiti u aproksimiranju derivacije
- Međutim, ako je npr. $f(x) = a + bx^2$, aproksimirana derivacija je jednaka $f_2'(x) = 2bx + bh$, dok je egzaktno rješenje $2bx \rightarrow$ ako je h mali, a b nije prevelik, možemo se približiti egzaktnom rješenju, no onda treba paziti na grešku zaokruživanja u oduzimanju f(x+h)-f(x)

PRVA DERIVACIJA POMOĆU TRI TOČKE

- Bolji pristup u slučaju kvadratne funkcije f(x) je primjena formule sa 3 koraka, gdje se računa derivacija sa obje strane oko izabrane točke x_0 (prema naprijed i prema nazad), i onda se usrednji rezultat
- Ponovno primjenimo Taylorov razvoj, ali sada za $x_0 \pm h$

$$f(x = x_0 \pm h) = f(x_0) \pm hf' + \frac{h^2 f''}{2} \pm \frac{h^3 f'''}{6} + O(h^4)$$

- Zapisano koristeći skraćenu notaciju:

$$f_{\pm h} = f_0 \pm hf' + \frac{h^2 f''}{2} \pm \frac{h^3 f'''}{6} + O(h^4)$$

- Usrednjeni izraz za derivaciju koristeći 3 točke (→indeks 3):

$$f_3' = \frac{f_h - f_{-h}}{2h} - \frac{h^2 f'''}{6} + O(h^3)$$

GREŠKE U NUMERIČKOM DERIVIRANJU

- u formuli za 3 točke, greška u vodećem redu derivacije ide sa $h^2 \rightarrow$ član $h^2 f'''/6$ naziva se *greškom odsijecanja* ("truncation error")

- radi se o grešci do koje dolazi jer je u izvođenju formalizma za račun derivacije Taylorov red odrezan, nisu uzeti svi članovi Taylorovog razvoja

- ključni uzroci greške u numeričkom modeliranju derivacije: a)greška odsijecanja i b)greška zaokruživanja

- oduzimanjem gornja dva izraza za $f_{\pm h}$ dobiva se izraz iz kojeg se može dobiti formula za proračun druge derivacije:

$$f_h - 2f_0 + f_{-h} = h^2 f'' + O(h^4)$$

NUMERIČKI PRORAČUN DRUGE DERIVACIJE

- Izraz za proračun druge derivacije funkcije f(x) koristeći metodu tri točke:

$$f'' = \frac{f_h - 2f_0 + f_{-h}}{h^2} + O(h^2)$$

- Može se definirati i formula za deriviranjem pomoću 5 točaka (x_0-2h, x_0-h, x_0, x_0+h, x_0+2h)
- Potreban je Taylorov razvoj u intervalu (-2h,2h) oko x_0:

$$f_{\pm 2h} = f_0 \pm 2hf' + 2h^2 f'' \pm \frac{4h^3 f'''}{3} + O(h^4)$$

- Izraz za prvu derivaciju pomoću 5 točaka:

$$f'_{5c} = \frac{f_{-2h} - 8f_{-h} + 8f_h - f_{2h}}{12h} + O(h^4)$$

(korisno npr. ako je funkcija polinom 4. reda)

PRVA I DRUGA DERIVACIJA FUNKCIJE

- Izrazi za prvu i drugu derivaciju funkcije mogu se napisati kao (→detaljan izvod u M.H. Jensen knjizi)

$$\frac{f_h - f_{-h}}{2h} = f_0' + \sum_{j=1}^{\infty} \frac{f_0^{(2j+1)}}{(2j+1)!} h^{2j}$$

$$\frac{f_h - 2f_0 + f_{-h}}{h^2} = f_0'' + 2\sum_{j=1}^{\infty} \frac{f_0^{(2j+2)}}{(2j+2)!} h^{2j}$$

- u oba slučaja, greška je veličine kao $O(h^{2j})$

PRIMJER NUMERIČKOG PRORAČUNA DERIVACIJE

- Dosad stečeno znanje o numeričkim metodama za deriviranje primjenit ćemo na primjeru modeliranja druge derivacije funkcije $f(x)=e^x$

- pritom ćemo ponoviti znanja o pisanju podataka u datoteke, korištenju funkcija u naprednom programiranju, prijenosu varijabli po referenci

- 1. funkcija učitava ulazne podatke (vrijednost x, početni korak h, broj koliko puta ćemo korak smanjivati za faktor 2)
- 2. funkcija računa drugu derivaciju
- 3. funkcija ispisuje rezultate u vanjsku datoteku
- radi preglednosti, deklaracije svih funkcija navodimo na početku programa (moguće je i u vanjskoj datoteci zaglavlja)

PRIMJER NUMERIČKOG PRORAČUNA DERIVACIJE

```
//     Program koji racuna drugu derivaciju funkcije exp(x); ulazni parameteri su
//   argument x, velicina koraka h, ukupan broj dijeljenja koraka h sa 2
# include  <iostream>
# include  <cmath>

void initialise (double *, double *, int *);
void second_derivative( int, double, double, double *, double *);
void output( double *, double *, double, int);
int main()
{
        // deklaracije varijabli
        int number_of_steps;
        double x, initial_step;
        double *h_step, *computed_derivative;

        //  ucitavanje ulaznih podataka
        initialise (&initial_step, &x, &number_of_steps);

        //  alociranje prostora u memoriji za 1D polja
        //  h_step and computed_derivative

        h_step =  new double[number_of_steps];
        computed_derivative = new double[number_of_steps];

        //  proracun druge derivacije od exp(x)

        second_derivative( number_of_steps, x, initial_step, h_step,
computed_derivative);

        //  Ispis rezultata

        output(h_step, computed_derivative, x, number_of_steps );
        // oslobadjanje memorije
        delete [] h_step;
        delete [] computed_derivative;
        return 0;
}   // end main program
```

PRIMJER NUMERIČKOG PRORAČUNA DERIVACIJE

```
//
//     Ucitavanje ulaznih podataka: pocetni korak, argument x, broj koraka
//

void initialise (double *initial_step,  double *x, int *number_of_steps)

{

        printf("Ucitaj pocetni korak, x i broj koraka\n");
        scanf("%lf %lf %d",initial_step, x, number_of_steps);
        return;

}  // end of function initialise
```

PRIMJER NUMERIČKOG PRORAČUNA DERIVACIJE

```
//  funkcija za proracun druge derivacije

void second_derivative( int number_of_steps, double x, double initial_step,
double *h_step, double *computed_derivative)

{
       int counter;
       double h;

       // korak se racuna u petlji


       h = initial_step;

       //  proracun za razlicite korake za deriviranje
        for (counter=0; counter < number_of_steps; counter++ )
        {

          //  korak i derivacija se pohranjuju u polje

             h_step[counter] = h;
             computed_derivative[counter] =
                         (exp(x+h)-2.*exp(x)+exp(x-h))/(h*h);
             h = h*0.1;
        }    // end of do loop

        return;

}   // end of function second derivative
```

PRIMJER NUMERIČKOG PRORAČUNA DERIVACIJE

```
//    funkcija za ispis rezultata

void output(double *h_step, double *computed_derivative, double x,
            int number_of_steps )

{

     int i;
     FILE *output_file;
     output_file = fopen("out.dat", "w") ;
     for( i=0; i < number_of_steps; i++)
       {
         fprintf(output_file, "%20.16E %20.16E \n",
                 log10(h_step[i]),log10(fabs(computed_derivative[i]-exp(x))/exp(x)));
        }
     fclose (output_file);

}  // end of function output
```

- ispis sadrži proračun relativne greške:

$$\epsilon = log_{10}\left(\left|\frac{f''_{\text{computed}} - f''_{\text{exact}}}{f''_{\text{exact}}}\right|\right)$$

REZULTATI PRIMJERA NUMERIČKOG PRORAČUNA DERIVACIJE

- Ispis izračunate vrijednosti funkcije e^x i egzaktno rješenje:

x	$h = 0.1$	$h = 0.01$	$h = 0.001$	$h = 0.0001$	$h = 0.0000001$	Exact
0.0	1.000834	1.000008	1.000000	1.000000	1.010303	1.000000
1.0	2.720548	2.718304	2.718282	2.718282	2.753353	2.718282
2.0	7.395216	7.389118	7.389057	7.389056	7.283063	7.389056
3.0	20.102280	20.085704	20.085539	20.085537	20.250467	20.085537
4.0	54.643664	54.598605	54.598155	54.598151	54.711789	54.598150
5.0	148.536878	148.414396	148.413172	148.413161	150.635056	148.413159

- smanjivanjem koraka za deriviranje h, numeričko rješenje se približava egzaktnom
- Međutim, daljnjim smanjivanjem h, greška numeričkog rješenja se počne povećavati!
- Postoji limit kako mali h može biti da se ne gubi na preciznosti

Zadatak 3:

Napisati program koji numerički računa prvu i drugu derivaciju funkcije $f(x)=e^x \sin(x)$ za x od -5 do 5 u koracima po 1, i za različite vrijednosti veličine koraka korištenog u deriviranju (npr. $h=0.1, 0.01, \ldots, 10^{-15}$).

Napisati odvojene funkcije i koristiti pozive po referenci u prijenosu podataka između različitih funkcija:

1) za proračun prve derivacije
2) za proračun druge derivacije

Treba izračunati i usporediti vrijednosti numeričke derivacije i analitičke derivacije navedene funkcije za vrijednosti x u gore zadanom intervalu. Treba izračunati relativnu grešku u proračunu prve i druge derivacije i nacrtati grafove ovisnosti greške o koraku h. Optimizirati program da ispisuje rezultate kada je greška minimalna.