Reševanje nelinearnih enačb, 1.del

Navedene strani so iz knjige *Octave z uvodom v numerične metode.*

Numerične napake

Rekurzivni izračun integrala: (str. 119)

Izračunali bomo vsoto

$$\sum_{n=0}^{15} \int_0^1 x^n \mathrm{e}^{-\lambda x}\, dx,$$

kjer je $\lambda = 0.54$. S pomočjo integriranja po delih dobimo rekurzivno formulo oblike $I_{n-1} = F(I_n)$:

$$\begin{aligned} I_n &= \int_0^1 x^n \mathrm{e}^{-\lambda x}\, dx = -\frac{1}{\lambda} x^n \mathrm{e}^{-\lambda x}\Big|_0^1 + \frac{n}{\lambda}\int_0^1 x^{n-1}\mathrm{e}^{-\lambda x}\, dx = -\frac{1}{\lambda}\mathrm{e}^{-\lambda} + \frac{n}{\lambda} I_{n-1} \\ I_{n-1} &= \frac{1}{n}\left(\lambda I_n + \mathrm{e}^{-\lambda}\right) \end{aligned}$$

Vsoto bomo izračunali na dva načina. Pri prvem načinu najprej izračunamo integral

$$I_0 = \int_0^1 \mathrm{e}^{-\lambda x}\, dx = -\frac{1}{\lambda}\mathrm{e}^{-\lambda x}\Big|_0^1 = \frac{1-\mathrm{e}^{-\lambda}}{\lambda},$$

druge pa z rekurzivno formulo in seštejemo. Pri drugem načinu pa privzamemo, da je vrednost nekega integrala, recimo I_{50}, ki je že ven iz območja, kjer te integrale seštevamo, enaka 0. Z rekurzivno formulo izračunamo integrale za druge $n < 50$ in seštejemo prvih 16 integralov. Ta način zakodiramo v proceduri `recint.m`.

```
% y = recint(lambda,n)
function y = recint(lambda,n)
    y = zeros(n+1,1);
    for i = n:-1:1
        y(i) = (lambda*y(i+1)+exp(-lambda))/i;
    end;
```

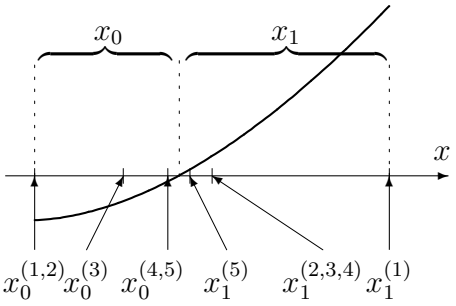
Izračunajmo vsoto na oba načina in jo primerjajmo s točno vrednostjo 2.31415. Opazimo, da smo z drugim načinom dobili točno vrednost, s prvim pa ne. To kaže na to, da je pomembno, v kakšnem vrstem redu izvajamo aritmetične operacije, saj zaradi nenatančne aritmetike lahko pride do velikih napak.

```
>> lambda=0.54;
>> z=zeros(1,16); z(1)=(1-exp(-lambda))/lambda;   % 1. nacin
>> for i=2:16 z(i)=((i-1)*z(i-1)-exp(-lambda))/lambda; end
>> vsota1=sum(z)
vsota1 = 2.94357
>> y=recint(lambda,50);   % 2. nacin
>> vsota2=sum(y(1:16))
vsota2 = 2.31415
```

Bisekcija

Poleg funkcijskega predpisa $f(x)$ je potrebno podati še krajišči začetnega intervala x_0 in x_1 ter natančnost ϵ. Začetni interval moramo izbrati tako, da bosta funkcijski vrednosti $f_i = f(x_i)$, $i = 0, 1$ v krajiščih različnega predznaka. (str. 143-146)

Na sliki je prikazano delovanje metode. Levo od rešitve so krajišča x_0, desno pa x_1. Zgornji indeks pomeni zaporedno številko iteracije: $x_0^{(4,5)}$ je levo krajišče intervala v četrti in peti iteraciji, $x_1^{(1)}$ pa desno krajišče v prvi iteraciji. Najprej izračunamo razpolovišče intervala. Nato vzamemo tisti podinterval, ki ustreza pogoju, da sta vrednosti funkcije v krajiščih različno predznačeni. Interval se v vsaki iteraciji razpolovi. Iteracijo ponavljamo toliko časa, dokler dolžina intervala ne pade pod predpisano natančnost. Kot rezultat vrnemo razpolovišče zadnjega intervala. Če v neki iteraciji zadenemo v razpolovišču točno vrednost, to vrnemo kot rešitev in bisekcijo zaključimo.



Prednosti in slabosti metode:

- Če je začetni interval pravilno izbran, metoda vedno konvergira.
- Če se na izbranem intervalu nahaja več ničel, ne moremo vedeti, h kateri od teh bo metoda konvergirala.
- Konvergenca je linearna (razmeroma počasna).
- Ničel, v katerih se znak funkcije ne spremeni, metoda ne zazna (dvojne ničle).

Algoritem:

- Izračunamo vrednosti $f_0 = f(x_0)$ in $f_1 = f(x_1)$. Če sta vrednosti različnega znaka, nadaljujemo, sicer javimo napako `error('bisekcija: f(x0)*f(x1) >= 0')`.
- Vstopimo v zanko.
- Izračunamo
$$x_2 = \frac{x_0 + x_1}{2} \quad \text{in} \quad f_2 = f(x_2).$$
- Če sta f_0 in f_2 različnega predznaka, postavimo $x_1 = x_2$ in $f_1 = f_2$, sicer pa $x_0 = x_2$ in $f_0 = f_2$.
- Če je $|f_2| < \epsilon$, zapustimo zanko in vrnemo rešitev x_2.
- Sicer ponovno vstopimo v zanko.

Algoritem zapišemo v proceduro `bisekcija.m`.

```
% [c,st] = bisekcija(f,a,b,e)
% Metoda bisekcije za resevanje enacbe f(x)=0 na intervalu [a,b].
% Vhod:
% f : funkcija
% a, b : krajisci intervala
% e : natancnost
% Izhod:
% c : resitev enacbe
% st : stevec stevila iteracij
function [c,st] = bisekcija(f,a,b,e)
  % izracun funkcijskih vrednosti
  fa = f(a);
  fb = f(b);
  % preverba pogoja obstoja nicle
  if fa*fb >= 0, error('bisekcija: f(a)*f(b) >= 0'), end;
  st = 0;
  % zanka se izvaja, dokler zaustavitveni pogoj ni izpolnjen
  while 1
    % izracun razpolovisca intervala
    c = (a+b)/2;
    fc = f(c);
    % izbor levega ali desnega podintervala glede na veljavnost pogoja
    if fa*fc < 0
      b = c;
      fb = fc;
    else
      a = c;
      fa = fc;
    end;
    % povecanje stevila iteracij
    st = st+1;
    % zaustavitveni pogoj:
    % ce je vrednost funkcije v razpoloviscu dovolj blizu 0, koncamo
    if abs(fc) < e, return, end;
  end;
```