Iskanje ničel funkcij z metodo bisekcije

Imejmo podano funkcijo f(x), ki ji želimo poiskati ničle, to je presečišča z x-osjo, kjer je vrednost f(x)=0. Včasih lahko ničle določimo analitično, pogosto pa ne. Tedaj moramo to narediti numerično, za kar obstaja več metod. Podrobneje si bomo ogledali metodo bisekcije ter tangentno metodo.

Pri vseh metodah za iskanje ničel je treba imeti oceno intervala, kjer ničle ležijo. Najlaže je, da si narišemo graf funkcije f(x) in iz njega določimo intervale z ničlami. Delali bomo z zveznimi funkcijami.

Osnove

Če je funkcija zvezna, ima na intervalu [a,b] vsaj eno ničlo, če je f(a)*f(b)<0.

Interval [a,b] razpolovimo: c=(a+b)/2 in izračunamo vrednost f(c).

Če je f(a)*f(c)<0, je ničla na intervalu [a,c], sicer pa je na intervalu [b,c].

Temu ustrezno v naslednji iteraciji premaknemo (zožimo) meje intervala, ki ga nato zopet razpolovimo. Postopek nato ponavljamo do želene natančnosti.

Če je ϵ_0=|b-a| začetna širina intervala in je ϵ želena toleranca določanja ničle, bomo za določitev ničle potrebovali

$n = \log_2 \frac{\varepsilon_0}{\varepsilon}$ korakov.

Naloga: določi ničli polinoma $f(x)=x^2+5x-1$

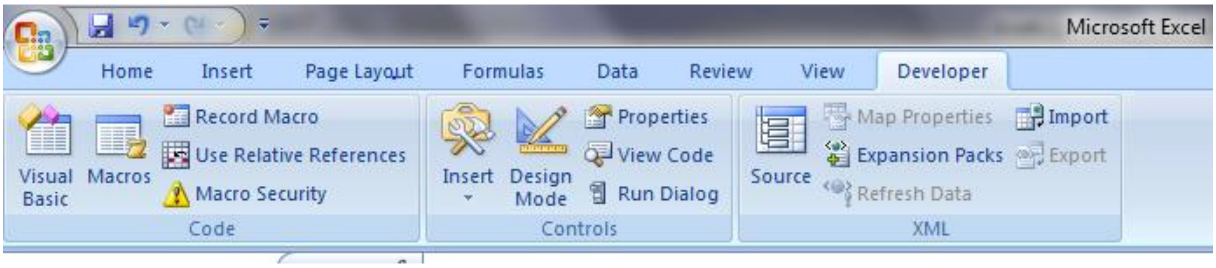
Postopek (=program) v Fortranu

```fortran
program bisekcija   ! zacetek programa
implicit none
real(8) :: a,b,c,f,fa,fb,fc,x,eps0,eps
integer(4) :: i     ! deklaracija (prijava) spremenljivk

eps=1.d-4           ! konvergencni kriterij
! na [a,b] je zagotovo samo 1 nicla; tega zato ne preverjamo

a=0.d0              ! zacetni vrednosti intervala
b=1.d0

do while (abs(b-a)>eps)  ! do while zanka
fa=f(a)         ! vrednost funkcije v levem krajiscu
fb=f(b)         ! vrednost funkcije v desnem krajiscu
c=(a+b)/2.d0    ! razpolavljanje intervala
fc=f(c)         ! vrednost funkcije v razpoloviscu

if(fa*fc<0)then ! nicla je med a in c
    b=c         ! premaknemo b
elseif(fb*fc<0)then ! ce ni med a in b, je pa med c in b
    a=c         ! premaknemo a
else
    print *, 'cestitamo, slucajno ste nasli ste niclo',c
endif

i=i+1           ! povecamo stevec za 1

end do          ! konec do while zanke

print *, 'nicla je', c  ! izpisemo niclo
end program bisekcija   ! konec programa

function f(x)   ! tule izracunamo vrednost funkcije f(x) v tocki x
implicit none
real(8) :: f,x  ! klasicna deklaracija
f=x*x+5*x-1     ! racun po formuli
return          ! konec racuna
end
```

Postopek v Excelu, 1.

Odpremo novo preglednico in najprej označimo posamezne stolpce, da bomo kasneje vedeli, kaj predstavljajo: v celico A1 vpišemo n(zaporedna številka iteracije), v B1 a (levo krajišče), ter nato še b (desno krajišče)v C1, x (razpolovišče) v D1,f(a)*f(x) v E1 in nazadnje navecja_napaka v F1.

Sedaj uporabimo t.i. Visual Basic, ki ga najdemo pod »Developer« zavihkom opravilnega traku:



Kliknemo, nato izberemo »Insert« in »Module« ter vnesemo naslednje funkcije:

```
Function f(x)
f = x * x + 5 * x - 1
End Function
```

Najprej sama funkcija f(x); po potrebi jo nadomestimo

```
Function x(a, b)
x = (a + b) / 2
End Function
```

Ta funkcija x razpolovi interval

```
Function novi_a(a, x, ff)
If (ff < 0) Then
  novi_a = a
Else
  novi_a = x
End If
End Function
```

Funkcija novi_a nadomesti levo krajišče intervala z vrednostjo x, če je produkt f(a)*f(x) negativen

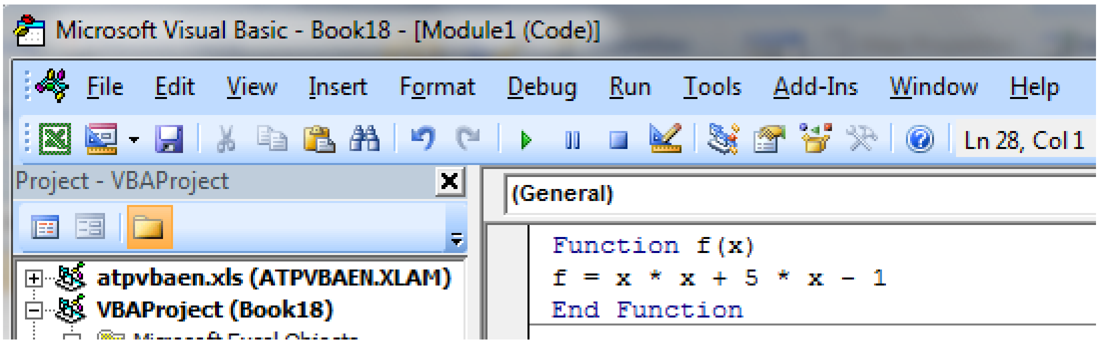
```
Function novi_b(b, x, ff)
If (ff < 0) Then
  novi_b = x
Else
  novi_b = b
End If
End Function
```

Funkcija novi_b nadomesti desno krajišče intervala z vrednostjo x, če je produkt f(b)*f(x) negativen

```
Function najvecja_napaka(a, b)
najvecja_napaka = Abs(b - a) / 2
End Function
```

Še največja napaka trenutnega približka za ničlo.

Nato kliknemo ikonico za Excel () v levem zgornjem kotu okna za Visual Basic (glej naslednjo sliko) in nadaljujemo s pisanjem v preglednico:



2. vrstico lahko pustimo prazno.

V B3 vnesemo levo krajišče intervala z ničlo (0), v C3 pa desno (1). V D3 nato vpišemo "=X(B3;C3)". V E3 vnesemo " =f(B3)*f(D3)", v F3 pa " =najvecja_napaka(B3;C3)".

V A4 vpišemo 2, v B4 pa " =novi_a(B3;D3;E3)". V C4 pride " =novi_b(C3;D3;E3)". E4 in F4 lahko s pomočjo miške nadaljujemo iz prejšnje vrstice.

Sedaj je 4. vrstica popolna in jo spet s pomočjo miške nadaljujemo po tabeli navzdol do željene natančnosti.

Postopek za Microsoft Excel in OpenOffice Calc, 2.

Odpremo novo preglednico ter v stolpce A-E vnesemo oznake a,b,(a+b)/2,f(a)*f(x),f(b)*f(x).

Drugo vrstico pustimo prazno.

V A3 vnesemo levo in v B3 desno krajišče začetnega intervala z ničlo funkcije.

V C3 vnesemo "=(A3+B3)/2", v D3 pa " =(A3*A3+5*A3-1)*(C3*C3+5*C3-1)", kar je produkt funkcije v levem krajišču in na sredini intervala. Podobno ponovimo v E3 za desno krajišče intervala "=(B3*B3+5*B3-1)*(C3*C3+5*C3-1)" .

V A4 pa vpišemo " =IF(E3<0; C3; A3)" in v B4 =IF(D3<0; C3; B3), kar je ekvivalentno prejšnjemu primeru s funkcijama "novi_a" in "novi_b", le da tu uporabimo Excelov vgrajeni IF stavek.

Ta ima naslednjo skladnjo oz. sintakso:

IF(pogoj; vrednost_if_true; vrednost_if_false)

Pogoj je logični izraz in ima lahko vrednost bodisi True bodisi False. Npr. a=b je lahko res ali pa ne; tretje možnosti ni. IF stavek vrne v celico, v kateri je napisan, *vrednost_if_true*, če je imel pogoj vrednost True, oziroma *vrednost_if_false*, če je imel pogoj vrednost False.

Pomen ukaza v celici A4 " =IF(E3<0; C3; A3)" je naslednji:

če je vrednost v celici E3<0 (vrednost pogoja bo True), se bo v celici A4 pojavila vsebina celice C3, če pa je E3≥0 (vrednost pogoja bo False), bo v A4 vsebina A3.

Sekantna metoda

Podobno kot pri metodi bisekcije moramo tudi tu poznati interval *[a,b]*, kjer se nahaja ničla funkcije. Tedaj je *f(a)*f(b)=0*. Pri sekantni metodi povežemo točki *(a,f(a))* in *(b,f(b))* s premico; presečišče te premice z abscisno osjo je naslednji približek ničli funkcije. Enačba premice se glasi:

$$k = \frac{f(b) - f(a)}{b - a} = \frac{y - f(a)}{x - a}$$

$$y = f(a) + \frac{f(b)-f(a)}{b-a}(x - a)$$

Ko *y=0*, je *x* enak

$$x = a - f(a)\frac{b - a}{f(b) - f(a)}$$

Imenujmo to absciso *c*. Sedaj izračunamo vrednost funkcije v tej točki, *f(c)*. Preverimo predznak produkat *f(a)*f(c);* če je ta negativen, prestavimo krajišče *b* v *c*, sicer pa *a* v *c*.

Pri tej metodi moramo za kriterij konvergence opazovati razdaljo med dvema zaporednima približkoma pravi ničli; c_i in c_{i+1}. Ko je $|\ c_i$ - $c_{i+}| < \epsilon$, s postopkom prenehamo.

Postopek za Microsoft Excel in OpenOffice Calc

C3 =(A3*A3+5*A3-1)

	A	B	C	D	E	F	G	H	I
1	a	b	f(a)	f(b)	c	f(a)*f(c)	f(b)*f(c)	abs(a_i-a_{i+1})	abs(b_i-b_{i+1})
2	f(x)=x^2+5x-1								
3	0	1	-1	5	0,166667	0,138889	-0,69444		
4	0,166667	1	-0,13889	5	0,189189	0,002536	-0,09131	0,166667	0
5	0,189189	1	-0,01826	5	0,19214	4,35E-05	-0,01192	0,022523	0
6	0,19214	1	-0,00238	5	0,192525	7,41E-07	-0,00155	0,002951	0
7	0,192525	1	-0,00031	5	0,192575	1,26E-08	-0,0002	0,000385	0
8	0,192575	1	-4,1E-05	5	0,192581	2,14E-10	-2,6E-05	5,02E-05	0

V A3 vnesemo levo krajišče začetnega intervala, v B3 pa desno. V C3 vnesemo '' =(A3*A3+5*A3-1)'',

V D3 '' =(B3*B3+5*B3-1)'', v E3 '' =A3-(B3-A3)*C3/(D3-C3)'', v F3 '' =(A3*A3+5*A3-1)*(E3*E3+5*E3-1)'' in v G3 '' =(B3*B3+5*B3-1)*(E3*E3+5*E3-1)''

V A4 vnesemo ukaz '' =IF(G3<0; E3; A3)''; če je vrednost v celici G3 negativna, bo v A4 prišla vsebina E3, sicer pa vsebina A3. V B4 pa vnesemo '' =IF(F3<0; E3; B3)''.

V stolpcih H in I spremljamo razliko med dvema zaporednima približkoma meja intervala; tako v H4 vnesemo '' =ABS(A4-A3)'' in v I4 =ABS(B4-B3). Smiselno je opazovati tisto vrednost, ki se spreminja. Ko pade razlika pod 0,0001, s postopkom prenehamo.

Z rumeno barvo (oz. rahlo osenčeno) so pudarjene spremembe programa glede na bisekcijo.

```
program sekantna   ! zacetek programa
implicit none
real(8) :: a,b,c,f,fa,fb,fc,x,eps0,eps,c0
integer(4) :: i     ! deklaracija (prijava) spremenljivk

eps=1.d-4           ! konvergencni kriterij
! na [a,b] je zagotovo samo 1 nicla; tega zato ne preverjamo
a=0.d0              ! zacetni vrednosti intervala
b=1.d0
i=1
c=a
c0=b

do while (abs(c-c0)>eps)  ! do while zanka
c0=c
fa=f(a)         ! vrednost funkcije v levem krajiscu
fb=f(b)         ! vrednost funkcije v desnem krajiscu
c=a-fa*(b-a)/(fb-fa)    ! razpolavljanje intervala
fc=f(c)         ! vrednost funkcije v razpoloviscu
write(*,'(7f11.8)')a,b,fa,fb,c,fc,abs(c-c0)
if(fa*fc<0)then ! nicla je med a in c
    b=c         ! premaknemo b
elseif(fb*fc<0)then ! ce ni med a in b, je pa med c in b
    a=c         ! premaknemo a
else
    print *, 'cestitamo, nasli ste niclo',c ! za vsak slucaj
endif
i=i+1           ! povecamo stevec za 1
end do          ! konec do while zanke
print *, 'nicla je', c  ! izpisemo niclo
print *, 'stevilo iteracij je', i
end program sekantna   ! konec programa

function f(x)   ! tule izracunamo vrednost funkcije f(x) v tocki x
implicit none
real(8) :: f,x  ! klasicna deklaracija
f=x*x+5*x-1     ! racun po formuli
return          ! konec racuna
end
```

Postopek v Excelu, 2.

Tudi tukaj lahko uporabimo Visual Basic makro ukaze, podobno kot pri bisekciji.

F3 f_x =x(B3;C3;D3;E3)

	A	B	C	D	E	F	G	H	I
1	n	a	b	f(a)	f(b)	x	f(a)*f(x)	x_novi-x_stari	
2									
3	1	0	1	-1	5	0,166667	0		
4	2	0,1666667	1	-0,13889	5	0,189189	0,031531532	0,022523	
5	3	0,1891892	1	-0,01826	5	0,19214	0,036350761	0,002951	
6	4	0,1921397	1	-0,00238	5	0,192525	0,036991642	0,000385	
7	5	0,1925247	1	-0,00031	5	0,192575	0,037075417	5,02E-05	
8									

```
Function f(x)
f = x * x + 5 * x - 1
End Function

Function x(a, b, fa, fb)
x = a - fa * (b - a) / (fb
- fa)
End Function

Function novi_a(a, x, ff)
If (ff < 0) Then
   novi_a = a
Else
   novi_a = x
End If
End Function

Function novi_b(b, x, ff)
If (ff < 0) Then
   novi_b = x
Else
   novi_b = b
End If
End Function

Function najvecja_napaka(a,
b)
najvecja_napaka = Abs(b -
a) / 2
End Function
```

Vnesemo ukaze v levem okvirčku in se vrnemo v preglednico.

V celice vpisujemo naslednje:

V A3 "1", vB3 "0" in v C3 "1".

V D3 vnesemo "=f(B3)" in v E3 "=f(c3)".

V F3 vnesemo " =x(B3;C3;D3;E3)" in v G3 " =B3*F3".

H3 ostane prazen.

V B4 vnesemo " =novi_a(B3;F3;G3)" in v C4 " =novi_b(C3;F3;G3)", medtem ko A2, D2, E2, F2 in G2 nadaljujemo iz prejšnje vrstice z miško. V H4 vnesemo "=ABS(F4-F3)"

Naslednje vrstice nadaljujemo z miško.

Tangentna metoda

Pri tej metodi nam načeloma ni potrebno poznati intervala z ničlo, pač pa si izberemo začetno točko *a* v bližini ničle in izračunamo funkcijsko vrednost v tej točki, *f(a)*. Poleg tega moramo poznati enačbo za odvod te funckije, *f'(x)*. Izračunamo vrednost odvoda v točki *a, f'(a)*. V točki *(a,f(a))* položimo tangento na krivuljo *f(x)*; smerni koeficient tangente je ravno *f'(a)*. Poiščemo presečišče tangente z x-osjo in ta točka nam služi kot naslednji približek ničle funkcije.

Enačba premice je: *y=f(a)+f'(a)*(x-a)*. Ko *y=0*, je *x* enak *a-f(a)/f'(a)*. Ta x je novi približek za ničlo, recimo mu a_1. Sedaj postopek ponovimo v tej točki. Tudi tukaj opazujemo absolutno vrednost dveh zaporedni približkov za ničlo; ko sta narazen za manj kot ϵ, s postopkom prenehamo.

Postopek v Excelu:

A4 | fx | 0

	A	B	C	D	E
1	Newtonova tangentna metoda				
2	f(x)=x*x+5*x-1		f'(x)*2*x+5		
3	x	y	y'	novi x	
4	0	-1	5	0,2	
5	0,2	0,04	5,4	0,192593	0,2
6	0,192593	5,49E-05	5,385185	0,192582	0,007407
7	0,192582	1,04E-10	5,385165	0,192582	1,02E-05
8					

Naj bo naša začetna točka x=0, kar vnesemo v celico A4. V B4 izračunamo funkcijsko vrednost po enačbi '' =A4*A4+5*A4-1''. V C4 izračunamo vrednost odvoda v točki A4: '' =2*A4+5'', v D4 pa novi približek za x: '' =A4-B4/C4''.

V naslednji vrstici vnesemo v A5 ''=D4'', B5, C5 in D5 nadaljujemo iz prejšnje vrstice z miško, v E5 pa vnesemo '' =ABS(A5-A4)''. S tem je vrstica A4 popolna in jo samo razpotegnemo navzdol.

Še program v fortranu:

```
program tangentna
implicit none
real(8) :: f,odvod,x0,xnovi,eps
integer(4) :: i

eps=1.d-4         !konvergencni kriterij
x0=0.d0           !zacetna tocka
xnovi=x0+2.d0*eps !da pridemo prvice cez do
while test
i=0
do while(abs(xnovi-x0)>eps)
   i=i+1          !stevec iteracij povecamo za 1
   x0=xnovi       !shranimo stari x
   xnovi=x0-f(x0)/odvod(x0)  ! novi x
print *,xnovi,f(xnovi),abs(x0-xnovi),i ! sprotni
izpis
!write(*,'(3f12.8,I3)')xnovi,f(xnovi),abs(x0-
xnovi),i
enddo

end program tangentna

function f(x)
implicit none
real(8) :: f,x
f=x**2+5.d0*x-1.d0
return
end

function odvod(x)
implicit none
real(8) :: odvod,x
odvod=2.d0*x+5.d0
return
end
```