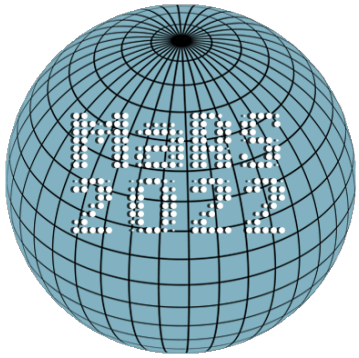
Numerično reševanje nelinearnih enačb

Domen Klopčič, Kanisaja Nika Kovačič, Adam Rakun
Mentor: Jakob Svetina



Povzetek

Ali ste vedeli, da računalnik ni samo za igranje igric? V našem projektu smo ugotovili, da je lahko zelo priročen pripomoček za matematike, saj lahko namesto nas opravi težko delo. Zato smo napisali nekaj algoritmov, po katerih nam računalnik najde rešitve nelinearnih enačb (in to s čisto majhno napako).

1 Uvod

Numerična metoda je postopek, s katerim iz numeričnih podatkov izračunamo numerični približek za rešitev nekega matematičnega problema.

Spoznali smo različne numerične metode, s katerimi lahko rešujemo nelinearne enačbe. Preden pa smo začeli z obravnavanjem le teh, smo definirali še nekaj pojmov, ključnih za nadaljno razumevanje.

Začeli smo pri enačbah in si na hitro ogledali, kaj sploh so nelinearne enačbe. Nato smo grafično in analitično definirali odvode. Kasneje pa smo si pogledali še različne vrste ničel in ključno neeačbo, s katero preverimo, ali

na določenem inervalu leži liho število ničel. S tem smo imeli dovolj osnove, da smo se lahko začeli ukvarjati z glavnim delom projekta - numeričnimi metodami.

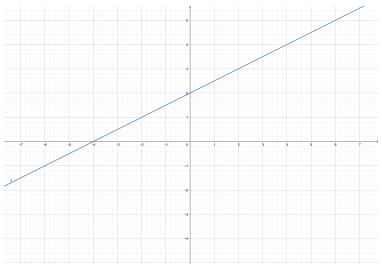
Prva numerična metoda, ki smo jo spoznali, je bisekcija. Temelji na večkratnem zaporednem razpolavljanju intervala, na katerem leži ničla. Vsakič ko interval razpolovimo, preverimo na katerem od novih intervalov je ničla. Slednje ponavljamo, dokler ne pridemo do dovolj majhnega intervala. Da tega ne počnemo predolgo, smo napisali algoritem, ki to stori namesto nas. Ker pa je bisekcija zelo preprosta metoda in ima tudi svoje slabosti, smo se premaknili na naslednjo metodo, iteracijo.

Iteracijska metoda najde ničlo s pomočjo določanja fiksnih točk. Za slednjo smo tudi spisali algoritem in se še kar nekaj časa zadržali z njo, saj je osnova zadnje metode, ki jo bomo omenjali v članku - Newtonove oziroma tangentne metode.

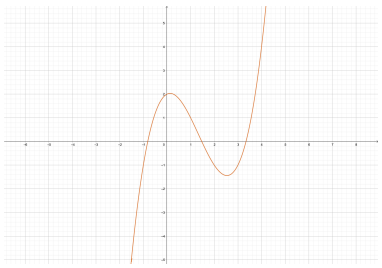
Newtnovo metodo smo najprej izpeljali grafično in nato še analitično. Ugotovili smo, da je od vseh metod, ki smo jih uporabljali, ta najbolj učinkovita. Algoritem zanjo ni zahteven, saj uporabi velik del iteracijskega algoritma.

2 Linearne in nelinearne enačbe

Definicija 1. ***Linearna enačba*** *(z eno neznanko) je enačba, ki jo lahko zapišemo v obliki* $kx+n=0$*, kjer sta koeficienta* k *in* n *poljubni realni števili. Če je* $k=0$ *in* $n=0$*, potem je rešitev linearne enačbe vsako realno število* x*. Če je* $k=0$ *in* $n\neq 0$*, potem je linearna enačba nerešljiva.*



Slika 1: Primer linearne funkcije z enačbo $f(x)=\frac{x}{2}+2$.



Slika 2: Primer nelinearne funkcije z enačbo $f(x)=\frac{x^3}{2}-2x^2+\frac{x}{2}+2$.

Definicija 2. ***Nelinearne enačbe*** *so algebrske enačbe druge ali višje stopnje.*

- *Nelinearna enačba druge stopnje se imenuje kvadratna enačba.*

- *Nelinearna enačba tretje stopnje se imenuje kubična enačba.*

- *Nelinearna enačba četrte stopnje se imenuje kvarčna enačba.*

- *Nelinearna enačba pete stopnje se imenuje kvintična enačba.*

Dokazano je, da ne obstaja analitična metoda za reševanje katere koli nelinearne enačbe stopnje 5 ali več.
Primer nelinearne enačbe z eno spremenljivko je $x^2 + 3x + 2 = 0$. Primera nelinearnih enačb tretje in četrte stopnje več spremenljivk sta $x^2+y^3+3xy = 4$ in $8yzx^2 + y^2 + 2z^2 + x + y + z = 4$.

3 Odvodi

Imamo poljubno krivuljo, podano s funkcijo $f(x)$. Zanima nas naklon te krivulje v neki izbrani točki A. Opazimo, da bo določitev naklona na krivulji težje opravilo, kot pri premicah.

Opazovanje krivulje omejimo na nek manjši interval. V tem intervalu si na krivulji izberemo še drugo poljubno točko B. Skozi točki narišemo premico. Določimo še točko C, ki ima isto ordinato kot točka A in isto absciso kot točka B. Naklon te premice je določen z razmerejem dolžin katet AC in BC v trikotniku ABC. Premaknimo točko B proti točki A. Ugotovimo, da se spreminja naklon premice. Ko s točko B dosežemo točko A, trikotnik izgine, premica pa postane dotikalnica (tangenta) na krivuljo v točki A. Ugotovimo, da je najboljša mera za naklon krivulje v točki A enaka naklonu tangente na krivuljo skozi točko A.

Položaj točke A v koordinatni mreži naj bo $(a, f(a))$. Ko smo z a označili absciso točke A, je njena ordinata $f(a)$, saj leži točka na krivulji. Položaj točke B naj bo $(b, f(b))$, točka C pa ima položaj $(b, f(a))$. Dolžina stranice CB je enaka $\Delta f = |f(b) - f(a)|$, dolžina stranice AC je enaka $\Delta x = b - a$, količnik dolžin, ki je mera za naklon, pa je enak $k = \frac{f(b)-f(a)}{b-a}$.

Ko približujemo točko B k točki A, se b približuje k a in $f(b)$ k $f(a)$, količnik k pa se približuje naklonu tangente. Temu postopku približevanja pravimo limitiranje, če se vrednost količnika k približuje nekemu končnemu številu, pa to število imenujemo limita. Zapišemo ga v obliki

$$k = \lim_{b \to a} \frac{\Delta f}{\Delta x} = \lim_{b \to a} \frac{f(b) - f(a)}{b - a}.$$

Vpeljemo novo oznako $h = b - a$. Potem je $b = a + h$. Ko gre b proti a, gre h proti 0. Izraz za velikost naklona se sedaj glasi

$$k = \lim_{h \to 0} \frac{\Delta f}{\Delta x} = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}.$$

Dobljeni količnik imenujemo **diferenčni količnik**. Podaja nam naklon tangente na krivuljo v točki A. Ker je tangenta premica, naklon krivulje v točki pa smo določili z naklonom tangente, to pomeni, da ima naklon poljubne krivulje enake lastnosti, kot jih ima za linearno funkcijo. Za naraščajoče funkcije bo pozitiven, za padajoče negativen. Za strme bo po absolutni vrednosti večji od 1 in za položne manjši od 1. Ker je po velikosti enak tangensu kota med premico in abscisno osjo, imamo s tem tudi podatek, pod kakšnim kotom tangenta na krivuljo seka abscisno os.

Primer za funkcijo $f(x) = x^2$. Pri odvajanju nelinearnih enačb eksponenti postanejo faktor, s katerim množimo spremenljivko, eksponent spremenljivke pa se zmanjša za 1. Velja $f'(x) = 2x$.

Če ima funkcija f v točki a odvod (torej je $f'(a)$ končno število), potem pravimo, da je funkcija f v točki a odvedljiva. Če ima funkcija f za vsako točko na intervalu $[a, b]$ končen odvod, potem pravimo, da je funkcija f odvedljiva na intervalu $[a, b]$.

4 Bisekcija

Bisekcija je metoda za iskanje ničel funkcije f. Preden pa metodo definiramo in pokažemo njen algoritem, moramo razumeti še dve stvari.

4.1 Ničle

Ničla funkcije je točka, v kateri funkcija seka abscisno os. Opiše jo enačba $f(x) = 0$. Poznamo več vrst ničel.

Prva ničla, s katero se srečamo, je enostavna ničla. Zanjo katero velja $f'(x) \neq 0$. To je ničla, kjer graf seka abscisno os pod določenim kotom. Primer take ničle najdemo v linearni enačbi.

Poznamo tudi m-kratno ničlo, ki je ničla nelinearne enačbe stopnje m, kjer je $f^{(m)}(x) \neq 0$ za $m \in \mathbb{N}$, odvodi nižjih stopenj pa so enaki 0. Te ničle delimo na dve vrsti.

Soda ničla je tista, kjer graf abscisne osi ne seka, vendar se je dotika. Funkcije, ki imajo take ničle, so tiste, ki imajo stopnjo $m = 2k$, kjer je $k \in \mathbb{N}$.

Liha ničla je ničla funkcije stopnje $m = 2k + 1$.

4.2 Prisotnost ničle

Prvi in ključni korak bisekcije je to, da ugotovimo ali je na danem intervalu sploh kakšna ničla.

Vzemimo interval $[a, b]$. Prisotnost ničel na tem intervalu preverimo z neenačbo

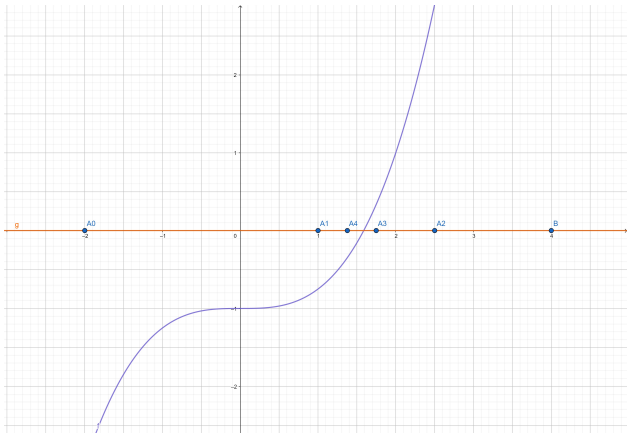
$$f(a) \cdot f(b) < 0.$$

S tem preverimo, če sta $f(a)$ in $f(b)$ na isti ali nasprotni strani abscisne osi. Če je njun zmnožek manjši od nič, pomeni, da sta na različnih straneh, saj imata nasprotna predznaka. Sledi, da graf na tem intervalu zagotovo seka abscisno os. Posledično je tu ničla.

Težava, s katero se srečamo, je, da ta način preveri le, če so prisotne lihe ničle in ne ničle nasploh.

4.3 Definicija bisekcije

Bisekcija je metoda za iskanje ničle funkcije f na intervalu $[a, b]$, za katero vemo, da je v robovih intervala nasprotno predznačena (velja $f(a) \cdot f(b) < 0$). Temelji na dejstvu, da ima taka funkcija na danem intervalu vsaj eno liho ničlo. V vsakem koraku metode razpolovimo trenutni interval ter naslednji korak nadaljujemo na desnem ali levem podintervalu. Izberemo glede na to, na katerem izmed podintervalov se nahaja ničla (



Slika 3: Primer bisekcije finkcije.

4.4 Algoritem bisekcije

```
1      function [a, b, stevec] = bisekcija_funkcija (f,a,b,e)
2      stevec = 0;
3      if sign (f(a)) == sign (f(b))
4      error ('f v tockah a in b ni nasprotno predznacena ');
5      while abs(b - a) > e
6      c = (a + b) / 2;
7      stevec = stevec + 1;
8      if sign(f(a)) == sign(f(c))
9      a = c;
10     else
11     b = c;
12     end
13     end
14     stevec
15     [a, b]
16     end
```

Algoritem prejme podatke f, a, b in e:

- f je funkcija, katere ničle iščemo,
- a je prvo krajišče intervala,
- b je drugo krajišče intervala,
- e je toleranca (dovoljena napaka pri približku ničle).

V algoritem vključimo števec, ki se z vsako ponovitvijo zanke `while` poveča za 1 in nam na koncu pove, kolikokrat se je bisekcija izvedla za dano napako. Algoritem najprej preveri, če sta predznaka funkcije v a in b enaka. Če sta, se program ustavi in vrne napako. Nato se požene zanka `while`, katere pogoj je, da je razdalja od a do b večja od tolerance. Če je pogoj izpolnjen, se izvede en korak bisekcije in števec se poveča za 1. V nasprotnem primeru se funkcija zaključi.

5 Iteracija

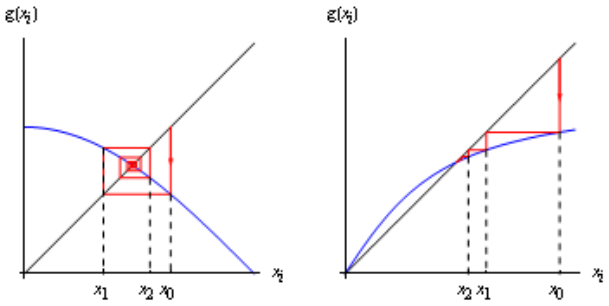
5.1 Definicija iteracije

Iteracija je metoda, s katero rešujemo nelinearno enačbo $f(x) = 0$. Najprej jo prepišemo v ekvivalentno obliko $x = g(x)$. Funkcijo g imenujemo iteracijska funkcija, saj rešitev enačbe iščemo z iteracijo

$$x_{r+1} = g(x_r),$$

kjer je r korak iteracije, za katerega velja $r \geq 0$ in $r \in \mathbb{N}$.

Kot prvi približek ničle vzemimo poljuben približek x_0. Če je g na določenem intervalu, ki vsebuje x_0, skrčitev, zaporedje $(x_r)_{r\geq 0}$ konvergira k fiksni točki iteracijske funkcije g, ki ustreza ničli funkcije f.



Slika 4: Grafični prikaz iteracije.

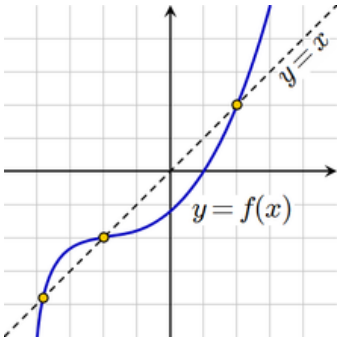
5.2 Fiksne točke

Fiksne točke ležijo na presečišču grafa funkcije f in premice $x = y$. Poznamo dve vrsti fiksnih točk.

Privlačna točka je točka, za katero velja neenačba $|g'(x)| < 1$. To pomeni, da se k tej točki z iteracijo bližamo.

Poznamo tudi *odbojno* točko, za katero velja nasprotno.

Do fiksne točke pridemo z iteracijo le v primeru, da začetni približek izberemo na intervalu, ki ga dobimo iz rešitve neenačbe $|g'(x)| < 1$.



Slika 5: Grafični prikaz fiksnih točk.

5.3 Algoritem iteracije

```
1      function x = iteracija (g,x0x,e,N)
2      stevec = 0;
3      while stevec < N
4      stevec = stevec + 1;
5      novi.x = g(x0);
6      if abs(novi.x - x0) < e
7      break;
8      end
9      x0 = novi.x;
10     end
11     x = novi.x;
12     stevec
13     end
```

Algoritem prejme podatke g, x_0, e in N:

- g je funkcija, katere fiksno točko iščemo,
- x_0 je začetni približek,
- e je toleranca (dovoljena napaka pri približku fiksne točke),
- N je število ponovitev (preprečuje, da bi se zanka while v primeru napačnega približka ponavljala v neskončnost).

V algoritem znova vključimo števec, ki se z vsako ponovitvijo zanke `while` poveča za 1 in nam na koncu pove, kolikokrat se je izvedla iteracija, preden je dosegla dano napako. Če števec doseže vrednost N, se program ustavi. V zanki `while` $novi.x$ dobi vrednost $g(x_0)$ in dokler absolutna vrednost razlike spremenljivk $novi.x$ in x_0 ni manjša od napake, se x_0 priredi vrednost $novi.x$ in ponovno vstopi v zanko. V nasprotnem primeru se program ustavi in izpiše števec.

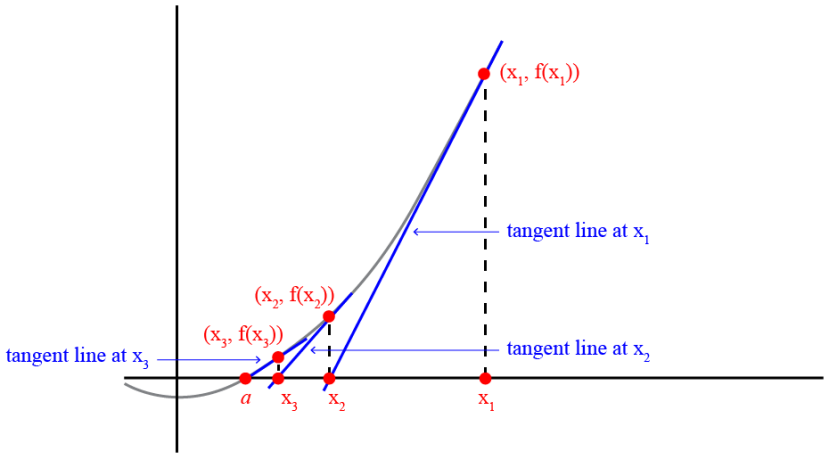
6 Tangentna metoda

6.1 Definicije tangentne metode

Tangentna metoda, drugače imenovana tudi **Newtonova metoda**, temelji na iskanju ničel funkcije f tako, da pripravimo ustrezno iteracijsko funkcijo za izvedbo navadne iteracije. Približek x_{r+1} za ničlo funkcije f dobimo iz približka x_r z zvezo

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}.$$

Geometrijsko x_{r+1} predstavlja presečišče abscisne osi in tangentne funkcije f v točki x_r.



Slika 6: Grafični prikaz tangentne metode.

6.2 Analitična izpeljava s Taylorjevo vrsto

Najprej smo tangentno metodo izpeljali s pomočjo Taylorjeve vrste. Pri izpeljavi smo za naš približek uporabili prva dva člena vrste, ostale pa zanemarili.

$$x_r = \text{trenutni približek}$$

$$x_r + h = \text{popravek približka}$$

$$f(x_r + h) = f(x_r) + h \cdot f'(x_r) + \dots$$

Ker želimo, da je funkcija v popravku približka enaka 0, smo to za izračun upoštevali.

$$0 = f(x_r) + h \cdot f'(x_r)$$

Iz dane enakosti smo izpostavili h in upoštevali še, da je h razlika med zaporednima približkoma x_r in x_{r+1}.

$$h = -\frac{f(x_r)}{f'(x_r)}$$

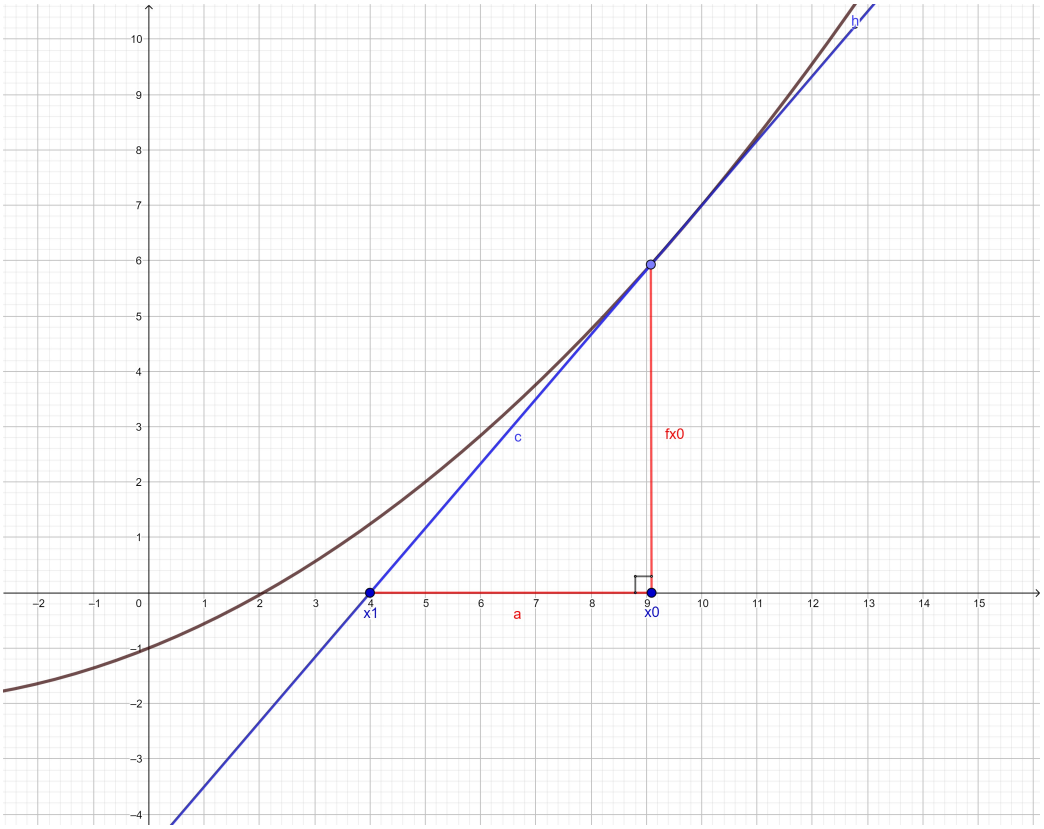
$$h = x_{r+1} - x_r$$

Nato smo le še izpostavili x_{r+1}.

$$x_{r+1} = x_r - \frac{f(x_r)}{f'(x_r)}$$

Dobili smo iteracijsko funkcijo, ki smo jo uporabili v našem algoritmu.

6.3 Grafična izpeljava



Slika 7: Grafični prikaz tangentne metode.

Najprej smo izpeljali, da je $f(x)$ enak koeficientu c, ki je enak $f'(x_0) \cdot a$.

$$f(x_0) = f'(x_0) \cdot a$$

Iz tega smo izpostavili a.

$$a = \frac{f(x_0)}{f'(x_0)}$$

V naslednjem koraku je podana definicija a, ki je enak razliki med dvema zaporednima približkoma ničle.

$$x_{r+1} + a = x_r$$

Izpostavimo popravek priblizka x_{r+1} in v enačbo vstavimo vrednost a. Dobili smo funkcijo tangentne metode.

$$x_{r+1} = x_r - a$$

$$x_{r+1} = x_r - \frac{f(x_0)}{f'(x_0)}$$

6.4 Algoritem Tangentne metode

```
1      function [x,X,k] = tangentna (f,df ,x0 ,tol ,N)
2      g = @(x) x - f(x)/df(x);
3      [x,X,k] = iteracija (g,x0 ,tol ,N);
4      end
```

Pri tem algoritmu potrebujemo poleg vhodov, ki jih sprejme algoritem iteracije, še odvod funkcije f, ki ga označimo z df. Nato samo definiramo iteracijsko funkcijo g in izvedemo algoritem iteracije.

Literatura

[1] J. Grošelj, *Pozdravljen, Matlab: osnove Matlaba za študente numerične matematike*, elektronski vir: https://www.fmf.uni-lj.si/~groseljj/matlab/pozdravljen_matlab, Ljubljana 2018.

[2] B. Plestenjak, *Razširjen uvod v numerične metode*, DMFA – založništvo, Ljubljana 2015.

[3] *Odvod funkcije*, v: Arnes, [ogled 27. 7. 2022], dostopno na http://www2.arnes.si/~vzagar/Sasa/_odvod/odvod10.htm.

Viri slik:

[4] *The General Iteration Method (Fixed Point Iteration Method)*, v: MathWorks, [ogled 28. 7. 2022], dostopno na https://www.mathworks.com/matlabcentral/fileexchange/69171-the-general-iteration-method-fixed-point-iteration-method?s_tid=blogs_rc_4.

[5] *Fixed point*, v: Academic Dictionaries and Encyclopedias, [ogled 28. 7. 2022], dostopno na https://en-academic.com/pictures/enwiki/70/Fixed_Point_Graph.png.

[6] *Newton's Method*, v: CalcWorkshop, [ogled 28. 7. 2022], dostopno na https://calcworkshop.com/derivatives/newtons-method/.