Gimnazija Kranj

NUMERIČNE METODE ZA ISKANJE PRIBLIŽKOV NIČEL FUNKCIJ

Matematika

Raziskovalna naloga

Avtor: Žan Ambrožič

Mentorica: Barbara Kušar, prof.

Kranj, 2023

Zahvala

Zahvaljujem se svoji profesorici matematike Barbari Kušar za pregled in nasvete glede popravkov ter izboljšav te raziskovalne naloge.

Kazalo vsebine

Kazalo grafov

Kazalo tabel

Povzetek

Iskanje ničel funkcij je zelo pomembno pri reševanju problemov v naravoslovnih znanostih in računalništvu, ampak ni vedno preprosto. Za linearno in kvadratno funkcijo imamo analitične metode, celo za polinome tretje stopnje obstaja Cardanova formula, vendar je precej dolga, velikokrat pa pri funkcijah nimamo nobene možnosti, da bi rešitev določili točno, zato uporabimo numerične metode, s katerimi se približamo ničlam funkcij na zadovoljivo natančnost. Poznamo več različnih numeričnih metod, v tej nalogi pa bom primerjal nekatere najbolj poznane glede na preprostost uporabe (računanje na papirju in z navadnim znanstvenim kalkulatorjem), uspešnost pri iskanju ničel in glede na število potrebnih korakov. Del primerjave bo kvalitativen, večina pa bo kvantitativna. Z uporabo računalnika je mogoče v nekaj sekundah poiskati približke za ničle več deset tisočkrat, kar omogoči uporabo velikega in reprezentativnega vzorca (v primerjavi z ročnim reševanjem, ki je počasno in bolj dovzetno za možnost človeške napake) za primerjavo metod. Med raziskovanjem sem razvil tudi svojo metodo, jo preizkusil in primerjal z ostalimi opisanimi na različnih vrstah polinomov.

Abstract

Finding the roots of functions is very important for solving problems in science and computing, but it's not always easy. We have analytical methods for linear and quadratic functions, even for third-degree polynomials there is Cardano’s formula, but it is quite long, and for most other functions we do not have any way of determining the solution exactly, so we use numerical methods to approximate the roots of the functions to a satisfactory accuracy. There are several different numerical methods, and in this paper, I will compare some of the most familiar ones in terms of ease of use (calculating on paper and with an ordinary scientific calculator), success in finding roots, and the number of steps required. Some of the comparison will be qualitative, but most will be quantitative. Using a computer, it is possible to find approximations to roots tens of thousands of times in a few seconds, allowing a large and representative sample to be used (compared to manual solving, which is slow and more susceptible to the possibility of human error) for comparing methods. In the course of my research, I have also developed my own method, which I tested and compare with others described on different types of polynomials.

Uvod

Matematika ni eksperimentalna znanost, kot sta fizika in kemija, vendar opredeljuje splošnejša dejstva in razmerja med števili ter je nepogrešljivo orodje v znanosti za napovedovanje in vrednotenje rezultatov poskusov. Ker je v matematiki s srednješolskim znanjem zelo težko odkriti kaj novega, sem se odločil, da se ne bom samo naučil nekaj novega in pisal o že znanih dejstvih, ampak bom tudi ocenil novo znanje in njegovo uporabo. Razlog je bila predvsem moja uvrstitev v državno ekipo za fizikalno olimpijado, na kateri se od tekmovalcev včasih zahteva numerično iskanje rešitev. Polinomske enačbe so v fiziki precej pogoste in brez grafičnega kalkulatorja je težko poiskati ničle polinomov višje stopnje (4 ali več), razen če so njihove ničle racionalne ter če so koeficienti in potence cela števila. Zato je edini način za pridobitev vrednosti približek. Med pripravami na tekmovanja iz fizike sem naletel na številne načine za aproksimacijo rezultata polinomov ali drugih enačb (trigonometrične funkcije, eksponentov, logaritmov itd.). Odločil sem se, da bom s pisalom, papirjem in navadnim znanstvenim kalkulatorjem (brez uporabe numeričnega reševalnika, ker je na tekmovanjih prepovedan) ocenil različne metode, v katerih primerih je z vsako od njih mogoče doseči dober približek in katera je za določeno vrsto enačbe najhitrejša. Pri tem predvidevam, da bo najučinkovitejša Newtonova metoda. Ker reševanje na roko ni preveč zanesljivo za učinkovitost točkovanja in traja dolgo časa, da dobimo le nekaj rezultatov, sem algoritme, ki sem jih želel preizkusiti, napisal v jeziku C# in jih izvedel na računalniku.

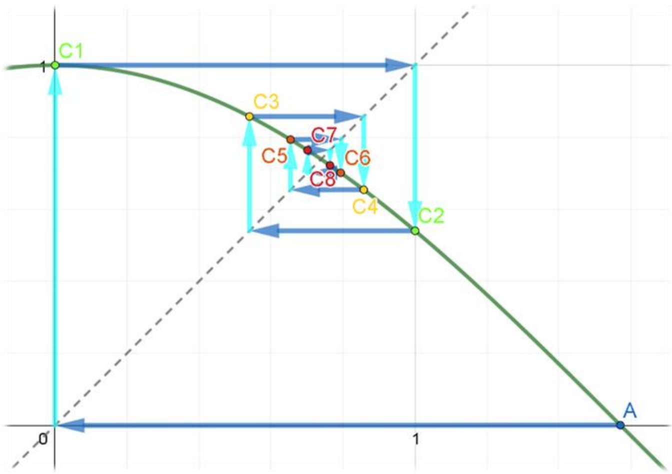
Teorija

Obstaja veliko metod za numerične aproksimacije, opisal pa bom le nekaj različnih, s katerimi sem se že srečal, ko sem se pripravljal na tekmovanja. Prva je navadna iteracijska metoda, sledita metoda bisekcije in sekantna metoda. Naslednja je Newton-Raphsonova metoda (znana tudi kot tangentna metoda), zadnji pa je Hornerjev algoritem, ki ga že dobro poznamo iz pouka matematike. Metoda "Regula-fasi" je bila izključena, saj je podobna sekantni metodi, le da so njene zahteve za začetne približke (ki jih ugibamo) bolj omejene (enake kot pri bisekciji). Na koncu bodo ovrednotene možnosti uporabe Taylorjeve vrste, kdaj bi lahko bila uporabna in s koliko členi. Med raziskavo sem zasnoval lastno metodo, ki bo prav tako preizkušena v tej raziskovalni nalogi.

V metodah 2-4 je na grafih uporabljena funkcija $f(x) = x^3 - 2x^2 + 1$ (na grafih izgleda "lepo" za potrebe predstavitve, ima tudi 2 iracionalni ničli, ki ju ni mogoče preprosto določiti z deljenjem ali kakšno drugo preprosto metodo), *C* bom uporabljal za označevanje približka (točka, *y*-vrednost bo izračunana, kadar bo njena vrednost pomembna v postopku (na primer za risanje tangente)) in x_n za *x*-koordinato *C*. V tej nalogi se bom osredotočil le na aproksimacijo realnih ničel (ne kompleksnih). Ničle (oz. približki ničel) izbrane funkcije $f(x)$ so -0,618, 1.000 in 1,618.

1. Navadna iteracijska metoda

Z navadno iteracijsko metodo se lahko približamo presečišču grafa funkcije in simetrale lihih kvadrantov $f(x) = x$, za uporabo moramo funkcijo najprej preurediti v obliko $x = f(x)$. Pri številnih enačbah lahko to storimo na več načinov (pri polinomih lahko izrazimo kateregakoli od ne-konstantnih členov, nato pa celotno enačbo razdelimo, da dobimo *x* (na prvo potenco) na eni strani, ali pa uporabimo koren ustrezne stopnje), vendar ima vsak način različno uspešnost pri iskanju približka rešitve s to metodo, poleg tega pa je odvisen tudi od ugibanega začetnega približka. Nekatere kombinacije vodijo do zelo hitre konvergence, nekatere pa lahko vodijo do divergence in sploh ne dajo rešitve. Najprej je potrebno izračunati vrednosti $f(x_n)$, kjer je x_n zadnji približek (ali začetni približek na začetku), nato pa metodo ponavljamo, dokler približki ne konvergirajo (ali divergirajo, kar pomeni slabo izbiro začetnega približka, napačno izražen izraz ali pa da te enačbe ni mogoče aproksimirati s to metodo). Metoda deluje tako, kot je prikazano na *Grafu 1*: izračunamo vrednost $f(x)$, nato pa poiščemo točko, v kateri je $y = x$, in nato se postopek ponovi. V nekaterih primerih presečišč premice $y = x$ in grafa $f(x)$ bodo aproksimacije konvergirale k presečišču iz ustreznega začetnega približka (kot je prikazano na *Grafu 1*), sicer pa bodo divergirale. Enačba za iteracijo je preprosto $x_{n+1} = f(x_n)$. Zelo znan primer je $x = \cos(x)$, ki ima samo eno rešitev in je vedno uspešna. Za začetni približek sem izbral $x = 1$ (*Graf 1*). Po teoremu je ta metoda uspešna, če je funkcija v izbranem intervalu zvezna in absolutna vrednost odvoda znotraj intervala ni večja od 1 (Yew, 2011), vendar sem s preizkušanjem našel nekaj enačb, ki konvergirajo k rešitvi, tudi če je začetni približek v točki, kjer je odvod manjši od -1. Funkcija $x = e^{\frac{1}{x}}$ konvergira k presečišču s simetralo lihih kvadrantov iz kateregakoli začetnega približka (razen od 0, ker tam ni definirana). Navadna iteracija deluje tudi z druge strani asimptote, čeprav funkcija vmes ni zvezna (ni definirana za $x = 0$). Odvod $f(x) = \sin^2(x) \cdot \cos(x) + x$ ponekod presega 1, vendar jo je mogoče zelo dobro aproksimirati s to metodo. Funkcije, za katero bi ta metoda divergirala v območju z absolutno vrednostjo odvoda manjšo od 1, nisem našel, kar podpira trditev. Za aproksimacijo ničel z navadno iteracijsko metodo je torej treba enačbo preurediti tako, da je njen odvod čim bližje ničli, saj se »položne« funkcije zelo malo spreminjajo in je potrebno majhno število aproksimacij. Če ni oblike enačbe z odvodom, ki bi bil od 0 oddaljen manj kot 1, še vedno obstaja

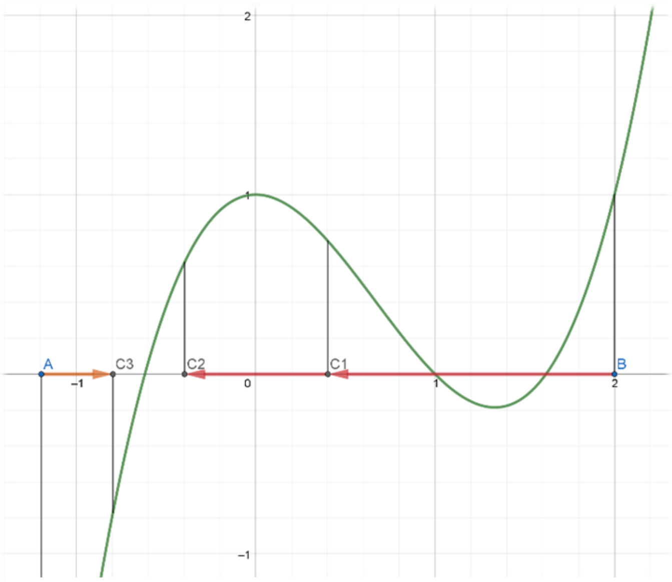


Graf 1: Primer uporabe navadne iteracije (prvih 8 korakov)

majhna možnost, da bo metoda delovala. V tem primeru obstajata dve možnosti - ali poskusimo vsako obliko z nekaj različnimi začetnimi približki, ali pa uporabimo kakšno drugo metodo.

2. Metoda bisekcije

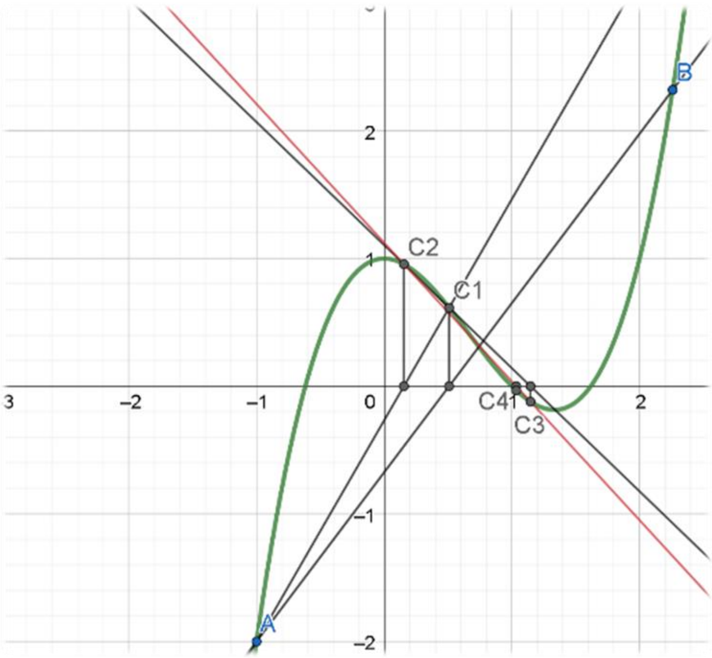
Ena najpreprostejših metod je bisekcija, ki na začetku zahteva dva začetna približka (ugibamo), *a* in *b*, ki predstavljata dve točki na grafu funkcije na nasprotni strani osi *x* - liho število sprememb predznaka zvezne funkcije med njima (projekciji na osi *x* sta na grafu označeni kot točki *A* in *B*). Aritmetično sredino začetnih približkov uporabimo kot prvi približek (*C1*) in za to točko izračunamo vrednost funkcije, vrednost *x* začetnega približka na isti strani osi *x* nato nadomestimo z aritmetično sredino *a* in *b*. Tako ostane en približek nad in eno pod osjo *x*. Postopek ponavljamo, dokler približek ne da vrednosti funkcije 0 (ali se ji dovolj približa), tam je ničla funkcije. Ta algoritem deluje za ničle lihe stopnje, saj zahteva spremembo predznaka (v primeru, da je ničla na prevoju, lahko dosežemo le lihe prevoje). Aproksimacijo z bisekcijo lahko zapišemo v obliki iteracije glede na prejšnji dve aproksimaciji (ali začetna približka):



Graf 2: Primer uporabe bisekcije (prvi 3 koraki)

$x_{n+1} = \frac{x_n + x_m}{2}.$

3. Sekantna metoda

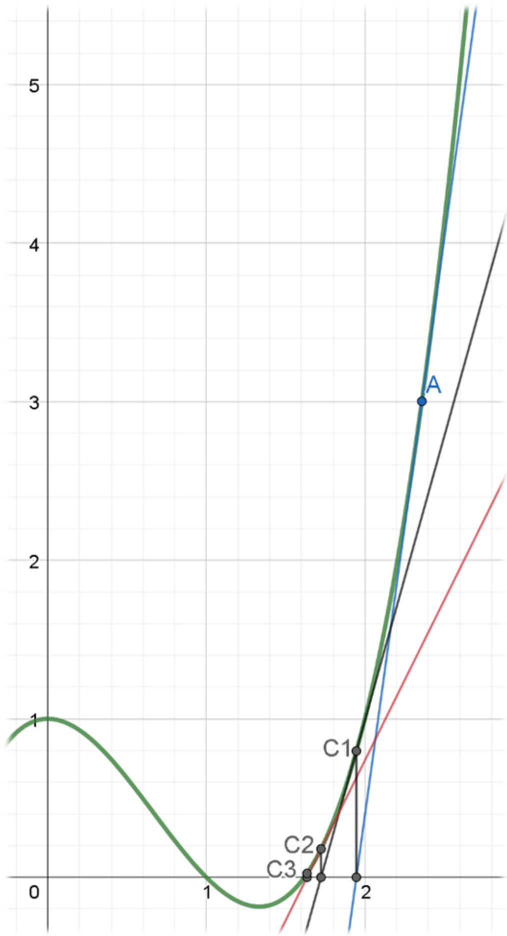


Graf 3: Primer uporabe sekantne metode (prvi 4 koraki)

Sekantna metoda je nekoliko naprednejša od metode bisekcije, pri kateri kot začetna približka izberemo dve točki na grafu funkcije, ki ne smeta imeti enake vrednosti y (sicer sekanta nikoli ne preseka osi *x*). Skozi obe točki potegnemo premico, njeno ničlo pa lahko enostavno določimo iz linearne enačbe in služi kot prvi približek. Točko na grafu izračunamo iz *x*-vrednosti tega približka in iz ene od točk začetnih približkov potegnemo novo premico skozi novo točko. Presečišče z vodoravno osjo se ponovno izračuna za naslednji približek in enak postopek ponovimo, dokler ne dosežemo ničle (ali dovolj dobrega približka). Vsak približek (točka) se uporabi dvakrat, nato se zamenja (enega od začetnih približkov uporabimo le enkrat, drugega pa dvakrat; v računalniškem algoritmu je bil dvakrat uporabljen tisti začetni približek z vrednostjo *y*, ki je bližje ničli, drugi je bilo zamenjan s prvim približkom). Potrebno število korakov za približevanje ničli je zelo odvisno od začetnih približkov. Naslednji približek lahko ponovno zapišemo v obliki iteracije, ki jo lahko enostavno izpeljemo iz linearne funkcije $y = kx + n$ (*k* je naklon in *n* je presečišče navpične osi, x_{n-1} označuje *x*-koordinato enega od začetnih približkov, kasneje ga nadomestimo z enim od približkov):

$$x_{n+1} = \frac{n}{k} = \frac{f(x_n) - k \cdot x_n}{k} = \frac{f(x_n)}{k} - x_n; k = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}.$$

4. Newton-Raphsonova (tangentna) metoda



Graf 4: Primer uporabe Newtonove metode (prvi 3 koraki)

Newton-Raphsonova metoda (v nadaljevanju Newtonova metoda) je znana kot najhitrejša in se zato pogosto uporablja v kalkulatorjih z algoritmi za reševanje enačb. Prav tako zahteva le en začetni približek v primerjavi z bisekcijsko in sekantno metodo, ki zahtevata 2. V točki začetnega približka narišemo tangento na graf funkcije, njeno presečišče z osjo *x* pa je *x*-koordinata prvega približka, nato v tem približku narišemo novo tangento na graf funkcije in na enak način dobimo naslednji približek, postopek pa ponavljamo, dokler približki ne konvergirajo k ničli. Metoda zagotovo odpove le v enem primeru, in sicer če je začetni približek natanko tam, kjer je (prvi) odvod funkcije enak 0 (lokalni minimum/maksimum/ prevoj), tangenta je vzporedna z osjo *x*, zaradi česar ni presečišča (razen če obstaja soda ničla, v katero smo postavili začetni približek, potem je ničla tam in metoda aproksimacije ni potrebna). Druga težava so lahko lokalni minimumi/maksimumi, ki lahko ujamejo približke (ponavljanje dveh podobnih približkov, dokler ni dosežena omejitev števila iteracij). Na prvi pogled se zdi, da so začetni približki v bližini lokalnih minimumov in maksimumov lahko problem, vendar njihova tangenta z vsakim približkom postane bolj »položna« in sčasoma (če ne takoj) je presečišče vodoravne dovolj daleč, da je naslednji približek na nekem popolnoma drugem delu grafa. Najtežji del aproksimacije je lahko iskanje odvoda, saj so nekatere funkcije zelo zapletene, pri polinomih pa odvajanje ne predstavlja resne težave. Pri metodi bisekcije in sekante je bilo treba izračunati enačbo premice, ker pa je bilo treba pri tej metodi izračunati že prvi odvod in vrednost funkcije v približku, standardni način z uporabo dveh točk morda ni najhitrejši. Tangento lahko izračunamo kot prva člena Taylorjeve vrste:

$$y(x) = f(c) + f'(c)(x-c) = f'(c)x + f(c) - f'(c)c$$

kjer *c* predstavlja *x*-koordinato približka. Presečišče tangente z vodoravno osjo je potem preprosto: $x_{n+1} = \frac{f'(x_n)x_n - f(x_n)}{f'(x_n)} = x_n - \frac{f(x_n)}{f'(x_n)}$. Zaradi tega je računanje približkov še hitrejše, ne le na računalniku ali grafičnem kalkulatorju, temveč tudi ročno in z osnovnim znanstvenim kalkulatorjem.

Če je odvod pretežko izračunati, še vedno obstaja alternativni način približka tangente, in sicer tako, da vzamemo točko v bližini približka in jo malo odmaknemo (za majhen *Δx*, namesto *dx*) ter nato zapišemo linearno enačbo iz teh dveh točk, vendar je ta postopek v primerjavi z načinom z odvodom veliko bolj zamuden in deluje podobno kot sekantna metoda.

5. Uporaba Taylorjeve vrste

Večino funkcij je možno zapisati v obliki polinoma z neskončnim številom členov (vrsta):

$f(x) = \sum_{n=0}^{\infty} \frac{\frac{d^n}{dx^n} f(s)}{n!} \cdot (x-s)^n$, kjer *s* označuje točko, v kateri je približek najnatančnejši. Manjša kot je razlika med *x* in *s*, večja je natančnost za točko z želeno *x* koordinato. Približek ničelnega reda ($n = 0$) ni funkcija *x*, prva aproksimacija je linearna, druga kvadratna ... Ne-polinomske funkcije lahko zapišemo kot polinome in ustrezne enačbe je potem včasih lažje rešiti za $f(x) = 0$. Na primer $\sin(x)$ lahko zapišemo približno okoli točke *s* kot $\sum \sin(s) + \cos(s)\,(x-s) - \frac{\sin(s)(x-s)^2}{2} - \frac{\cos(s)(x-s)^3}{6} + \cdots$, ker pa je vsak 4. odvod sinusa enak, lahko zapis malo poenostavimo:

$$\sum_{n=0}^{\infty} \left(\sin(s) \cdot \left(\frac{(x-s)^{4(n-1)}}{(4(n-1))!} - \frac{(x-s)^{4\left(n-\frac{1}{2}\right)}}{\left(4\left(n-\frac{1}{2}\right)\right)!} \right) + \cos(s) \cdot \left(\frac{(x-s)^{(4n-3)}}{(4n-3)!} - \frac{(x-s)^{(4n-1)}}{(4n-1)!} \right) \right).$$

Maclaurinova vrsta je posebna vrsta Taylorjeve, le da privzamemo $s = 0$, v tem primeru se zgornja enačba poenostavi še naprej: $\sum_{n=0}^{\infty} \left(\frac{x^{(4n-3)}}{(4n-3)!} - \frac{x^{(4n-1)}}{(4n-1)!} \right) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} \cdot x^{2n+1}$ in daje dobre približke okoli koordinatnega izhodišča.

Tabela 1: Približki sin(x) in cos(x) z Maclaurinovo vrsto - doseg približka znotraj natančnosti 0,01 (absolutno - zelena; relativno - modra), enote so radiani

Interval natančnosti *x* (±) - sin		Red vrste		Interval natančnosti *x* (±) - cos	
0.01	0	0	0, 1	0.14 (± 2·*k*·*π*) (*y* = 1)	
0.39	0.24	1, 2	2, 3	0.70	0.66
1.04	1.01	3, 4	4, 5	1.40	1.19
1.76	1.76	5, 6	6, 7	2.13	1.51
2.50	2.40	7, 8	8, 9	2.88	2.86*
3.25	2.87	9, 10	10, 11	3.62	3.59
4.00	3.88*	11, 12	12, 13	4.38	4.17
≈28.6	≈28.0	77, 78	78, 79	≈28.9	≈28.9
≈29.3	≈29.2*	79, 80	80, 81	≈29.7	≈29.4
*Blizu oddaljene ničle relativna napaka hitro preseže mejo > 1%					

Če je potrebujemo natančnost sinusa 0,01, so zanesljiva območja *x* v *Tabeli 1*. Linearni in kvadratni približek sta enaka, saj je drugi odvod sinusa samo negativni sinus, ki je v izhodišču enak 0 in dobimo $\frac{0 \cdot x^2}{2}$. Uporaben je večinoma le prvi približek ($\sin(x) \approx x$), saj ga je dovolj enostavno izračunati. Pri funkciji kosinusa je linearni približek enak približku ničelnega reda, saj so v razširitvi zastopane samo sodi odvodi (vsak drugi odvod je sinus, katerega vrednost v izhodišču je 0). Najpogosteje uporabljan približek je $\cos(x) \approx 1 - \frac{x^2}{2}$, katerega ničlo lahko najdemo s formulo. Ta približek je zelo natančen tudi pri kotih, večjih od 0,52. Z vsakim novim (neničelnim - drugim) členom se natančno območje poveča za približno 0,75 radiana. Vendar se širina natančnega območja Taylorjeve vrste določene stopnje spreminja z začetno točko (*s*), na primer natančno območje e^x je širše za majhne (tudi negativne) vrednosti *s*, kot pa za večje (kjer je veliko ožje). Širina natančnega območja ni konstantna niti za sinus in kosinus.

Polinome s Taylorjevo vrsto lahko tudi "poenostavimo", da opišemo obnašanje funkcije v okolici izbrane točke. Linearni približek je preprosto tangenta, na katero se opira Newtonova metoda. Kot je razvidno iz primerjave natančnosti Maclaurinovih vrst za sinus in kosinus v *Tabeli 1*, več členov prispeva k večji kakovosti in širšemu obsegu približkov enačbe, zato bi morda Taylorjeva vrsta s kvadratnim približkom deloval hitreje, saj obstaja enačba za ničle parabole in lahko hkrati aproksimira položaje dveh ničel. Da bi to poskusili, moramo najprej določiti enačbo za iteracijo.

$$f(x) \approx f(s) + f'(s)(x-s) + \frac{f''(s)(x-s)^2}{2} =$$
$$= \frac{f''(s)}{2} \cdot x^2 + (f'(s) - f''(s) \cdot s) \cdot x + f(s) - f'(s) \cdot s - \frac{f''(s) \cdot s^2}{2}$$

Sedaj imamo enačbo parabole v obliki: $y = a \cdot x^2 + b \cdot x + c$, ničle pa lahko najdemo s kvadratno enačbo: $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$. Vendar ima ta metoda težavo, če parabola nima nobenih realnih ničel, kar se lahko zgodi če je začetni približek daleč od vseh ničel, vendar se ne more pojaviti med dvema ničlama, če je funkcija med njima zvezna in sta najbližji lokalni minimum in maksimum na nasprotni strani osi *x*, saj se pred spremembo smeri (ko se spremeni znak drugega odvoda, se parabola obrne iz konveksne v konkavno ali obratno) parabola izravna in postane premica (pravzaprav tangenta na graf, kar je razvidno iz enačbe Taylorjeve vrste) pri eni določeni vrednosti x med vsakim parom zaporednih ničel. Na tej točki mora biti drugi odvod enak 0, za kar je znano, da se zgodi na prevojih (če se to zgodi med lokalnim minimumom in zaporednim maksimumom ali obratno). Ob upoštevanju tega sledi algoritem za iskanje ničel:

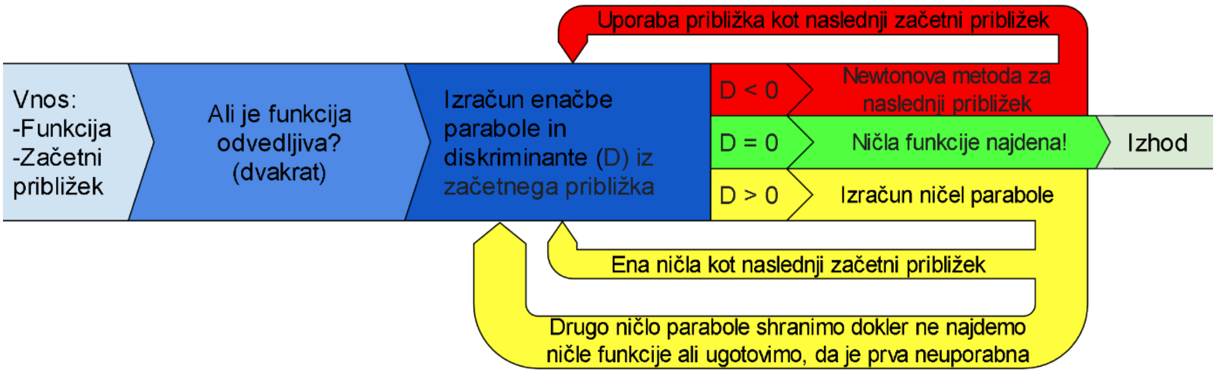


Diagram 1: Približki s Taylorjevo vrsto druge stopnje - "Kvadratna Newtonova" metoda

Ko najdemo 1 ali 2 ničli funkcije, želimo najti še ostale, takrat je treba spremeniti začetni približek, in če sumimo na robno ničlo (najnižja oz. najvišja ničla glede na *x*-os, kadar le ena od ničel parabole konvergira naprej – proti ničli funkcije), je verjetno najboljša možnost, da se oddaljimo od ničle parabole, ki ni konvergirala k ničli funkcije. Vrednost začetnega približka je treba premakniti dovolj daleč, da se parabola obrne – mora se spremeniti znak drugega odvoda. Vendar mnoge funkcije niso zvezne ali vsebujejo zaporedne lokalne minimume in maksimume na isti strani vodoravne osi, kar lahko poslabša napoved smeri naslednjega začetnega približka in zmanjša učinkovitost metode. Moja hipoteza je, da lahko ta metoda preseže Newtonovo metodo, vendar le pri nekaterih funkcijah s posebnimi značilnostmi. Glede na način delovanja jo bom imenoval "kvadratna Newtonova metoda".

6. Hornerjev algoritem

Ko polinom delimo z drugim polinomom, lahko dobimo ostanek, zaradi česar je enačbo običajno težje rešiti, če pa je ostanek enak 0, dobimo dva polinoma, ki ju lahko pomnožimo in dobimo nazaj prvotni polinom. Če linearno enačbo uporabimo za delitelja, lahko število členov v deljenem polinomu zmanjšamo enega za drugim, če so ničle znani (torej so ostanki enaki nič). Z deljenjem z linearnim polinomom lahko operacijo izvedemo precej enostavno in učinkovito. To pa zato, ker lahko v skladu s Hornerjevim pravilom vsak polinom zapišemo v obliki:

$$a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^2 = a_0 + x \cdot \Big(a_1 + x \cdot \big(a_2 + x \cdot (\cdots + x \cdot (a_{n-1} + x \cdot a_n) \cdots)\big)\Big)$$

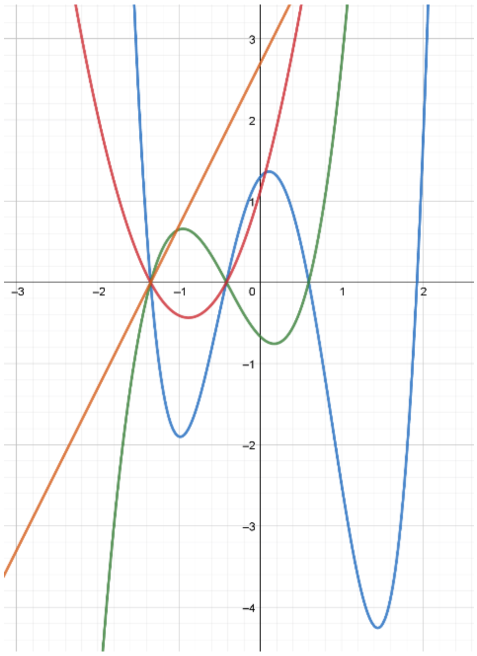
$x_0 = -1$	$a_4 = 1$	$a_3 = 0$	$a_2 = -2$	$a_1 = 3$	$a_0 = -4$
		$b_4 x_0 = -1$	$b_3 x_0 = 1$	$b_2 x_0 = 1$	$b_1 x_0 = -4$
	$b_4 = 1$	$b_3 = -1$	$b_2 = -1$	$b_1 = 4$	$b_0 = -8$

Slika 1: Določanje konstant s Hornerjevim algoritmom (He & Shiue, 2012)

Pri ničli (x_0), lahko zamenjamo konstanto z izrazom $a_m + a_{m+1} \cdot x_0 = b_m$, ki zadovoljuje enačbo. Te konstante dobimo po metodi, prikazani na *Sliki 1*. Če je $b_0 = 0$, je x_0 ničla polinima, ostale konstante $(b_n; n > 0)$ pa predstavljajo zaporedne koeficiente polinoma, ki ima identične ničle kot izvirni, le da sedaj ena manjka – tista s katero smo polinom delili.

Na ta način se stopnja polinoma zmanjša vsakič, ko najdemo eno od ničel. Kadar ostane kvadratna ali linearna enačba, je lahko analitično iskanje ničel hitrejše od ugibanja in ponavljanja metode.

Če namesto vsakega *x* vstavimo $\frac{c}{d}$ v prvotni polinom in množimo z d^n, kjer je *n* najvišja stopnja *x*, so vsi členi razen prvega (vodilnega, ki je vseboval $\frac{c^n}{d^n} \cdot d^n$) in zadnjega (konstanta) deljivi z obema številoma, *c* in *d*, vodilni koeficient je deljiv le s *c*, konstanta pa le z *d*. Za iskanje ničle $\frac{c}{d}$, so možni ulomki $\pm\frac{delitelj\ konstantnega\ člena}{delitelj\ vodilnega\ koeficienta}$. Z vsakim od teh lahko poskusimo najti racionalne ničle polinoma, vendar je ta metoda precej omejena in za daljše polinome z visokimi koeficienti (zlasti prvega in zadnjega) zelo zamudna. Prav tako zahteva celoštevilske koeficiente (da so deljivi). Učinkovitejša metoda je kombinacija deljenja z Newtonovo metodo. Po ugibanju začetnega približka se za aproksimacijo prve ničle uporabi Newtonova metoda (ki ne sme biti med dvema ničlama), nato polinom delimo z uporabo tega približka kot ničle in tako dobimo drug polinom z malo ali nič netočnosti v koeficientih. Dobljeno ničlo se nato uporabi kot naslednji približek za Newtonovo metodo in postopek se ponavlja, dokler ne dobimo vseh ničel (zadnji dve lahko najdemo analitično). Tako je iskanje vseh ničel polinoma veliko učinkovitejše od naključnega ugibanja in poskusov prehajanja lokalnega minimuma ali maksimuma, da bi se približali naslednji ničli. Seveda lahko uporabimo tudi druge prej opisane metode, vendar je znano, da je ta metoda najhitrejša (ker polinome zlahka odvajamo ni težav z uporabo Newtonove metode). Ker so pogoj le celoštevilske potence *x* (koeficienti so lahko realna števila), je to še en vidik, v katerem ta metoda prekaša tisto s poskušanjem možnih racionalnih ničel. Pravzaprav ta metoda ni omejena le na polinome, saj lahko večino funkcij na določenem območju opišemo z uporabo prvih nekaj členov Taylorjeve (ali Maclaurinove) vrste.



Graf 5: Zmanjševanje stopnje polinoma s Hornerjevim algoritmom

Postopek primerjave

Naključne enačbe

Glede na nastavitve je bilo ustvarjeno želeno število enačb z naključno izbranimi celoštevilskimi koeficienti. Da bi se izognili polinomom brez realnih ničel ali ničel v prevojih, je bil v vsaki enačbi vedno konstanten člen, v primeru, da je bil člen najvišje stopnje sod in je imel enak znak (+/-) kot konstantni člen, pa je bil znak vodilnega koeficienta obrnjen (z algoritmom).

Začetni približki (umetno ugibanje)

Začetni približki so bili generirani avtomatsko in povsem naključno, v enem setu je algoritem aproksimacije začel iz istega niza začetnih približkov za vse metode in enačbe.

Preizkušanje

Za primerjavo sem za vsak algoritem uporabil enak niz enačb in enak niz začetnih približkov. Algoritem se je ustavil, ko je bilo število, zaokroženo na določeno število decimalnih mest (*natančnost*), dvakrat zaporedoma enako, rezultati so bili zabeleženi. Drugi pogoj za ustavitev algoritma je bilo preveč iteracij, meja je bila določena kot $N_{stop} = 10^{natančnost}$, v tem primeru je algoritem zabeležil neuspeli poskus.

Zaradi preprostosti pri iteraciji s fiksno točko nismo iskali ničel $f(x)$, temveč rešitve za $f(x) = x$, saj je načinov preoblikovanja enačbe več in niso vsi enako dobri za to metodo, tako pa so enačbe ostale takšne, kakršne so bile.

Kvadratna Newtonova metoda je bila uporabljena samo s paraboličnim približkom, brez drugih korakov, kot je Newtonova metoda (normalna, linearna), negativna diskriminanta je bil šteta kot neuspešen poskus.

Pri vsaki najdeni ničli je bila preverjena vrednost funkcije, če je bilo odstopanje od 0 zunaj meja natančnosti, je bil zabeležen neuspeli poskus.

Obdelava podatkov

Program je poročilo shranil v datoteko *.txt* z razmiki (tabulatorji), ki pravilno razporedijo podatke v celicah, če se jih kopira v Excel. Enostavno je mogoče preveriti in izračunati uspešne poskuse in povprečno število iteracij (v primeru uspešno najdene ničle). Ker je racionalne funkcije težje napovedati (oblika in obstoj ničel), sem vse rezultate pregledal in enačbe z vsemi neuspešnimi poskusi narisal v Desmosu, da bi preverili, ali realne ničle sploh obstajajo. Če ničel ni bilo, je bila enačba odstranjena iz končnih izračunov. Ta postopek je bil izveden ločeno za Newtonovo metodo in metodo s fiksno točko (ker ima druga metoda drugačna merila, in sicer ne ničel, temveč presečišča z $y = x$).

Za namen primerjave lahko definiramo merilo dela – povprečno število iteracij za pridobitev ničle iz vsakega začetnega približka (ali 2): $Merilo\ dela = \frac{povprečno\ število\ iteracij}{povprečen\ uspeh\ pri\ iskanju}$. Manjša vrednost pomeni učinkovitejšo metodo (v smislu števila izračunov za rezultat). Upoštevati je treba, da število korakov za vsako iteracijo ni upoštevano, saj ni mogoče izmeriti, koliko več časa na primer traja odvajanje funkcije kot izračun njene vrednosti v neki točki ali seštevanje dveh vrednosti.

Primerjava

Uporaba

Tabela 2: Primerjava metod (splošno)

Metoda	Omejitve, pogoji	Prednosti
Navadna iteracijska metoda	-1 začetni približek za 1 ničlo, niso vsa primerna -Ni vsaka funkcija primerna -Funkcija mora biti zapisana v drugi obliki, za kar ni standardnega postopka	+Preprosta
Bisekcijska metoda	-2 začetna približka za 1 ničlo, na nasprotnih straneh *x*-osi -Konvergira počasi	+Preprosta +Vedno deluje
Sekantna metoda	-2 začetna približka za 1 ničlo	+Dokaj preprosta +Hitrejša od bisekcijske +Skoraj ni napačnih začetnih približkov
Newtonova metoda	-1 začetni približek za 1 ničlo -Odvedljiva funkcija (vsaj enkrat)	+Hitra +Skoraj ni napačnih začetnih približkov
Kvadratna Newtonova metoda	-1 začetni približek za več ničel -Odvedljiva funkcija (vsaj dvakrat) -Včasih potrebna Newtonova metoda	+Zelo hitra +Skoraj ni napačnih začetnih približkov (pomoč Newtonove metode) +Včasih najdemo 2 ničli
Horner (rac. ničle)	-Polinomi s celoštevilskimi koeficienti in stopnjami -Poskušanje možnih racionalnih ničel	+Ni ugibanja +Najdemo vse racionalne ničle
Horner-Newton	-1 začetni približek (mora biti pravi) -Odvedljiva funkcija (vsaj enkrat)	+Hitra +Najdemo vse realne ničle +Deluje za večino funkcij (pomoč Taylorjeve vrste)

Koraki

L predstavlja stopnjo polinoma; *n* je število ponovitev algoritma (le spodnjem primeru pri Newtonovi metodi v Horner-Newtonovem algoritmu, saj Hornerjev algoritem spremeni polinom tolikokrat, kolikor je potrebno za zmanjšanje njegove stopnje do kvadratne enačbe, odvod vsakega od teh polinomov je treba izračunati in določiti vrednost polinoma (*f(x)*) in njegove odvode (*df/dx*) v določeni točki). Seštevanje, odštevanje, množenje, deljenje in potenciranje števila štejejo kot "običajne operacije" (določanje vrednosti polinoma v določeni točki ni vključeno, ker so polinomi različnih dolžin, namesto tega se za to operacijo uporablja kategorija *f(x)*). *Iteracija ANS* predstavlja možnost ponavljanja izračuna z znanstvenim (ne-grafičnim) kalkulatorjem, pri čemer namesto dejanske vrednosti vpišemo *ANS* (prejšnji odgovor) in nato večkrat pritisnemo *ENTER*. Zaradi tega je metoda veliko hitrejša za uporabo, ko je enačba enkrat vnesena v kalkulator. Hornerjev algoritem zahteva deljenje polinoma, ki ga ni mogoče izvesti na znanstvenem kalkulatorju (neprogramabilnem).

Tabela 3: Operacije po metodah

Metoda	*f(x)*	*df/dx*	Običajne operacije	Iteracija ANS
Navadna iteracijska	1	0	0	Da
Bisekcijska	2	0	2	Ne
Sekantna	2	0	6	Ne
Newtonova	2	1	3	Da
Kvadratna Newtonova	3 (+ 2)	2	17 (+ 2)	Ne
Hornerjeva (rac. ničle)	0	0	$1 + 2(L - 1)$	/
Horner-Newtonova	$2n$	n	$1 + 2(L - 1) + 3n$	/

Zmogljivost

OPOMBA: Vse vrednosti so izračunane v programu Excel in so zaokrožene, vendar so med izračuni uporabljene točne vrednosti. V oklepajih ob kategoriji v krepkem tisku so nastavitve za algoritem: število členov, obseg stopenj členov, obseg koeficientov, število natančnih decimalnih mest, število umetno ugibanih začetnih približkov na enačbo ter število nizov in enačb na niz.

Preprosti polinomi (2-5 členov; stopnje 0-5; koeficienti (-5)-5; natančnost 3; 20 začetnih približkov/enačbo; 3 x 100 enačb)

Tabela 4: Zmogljivost metod: preprosti polinomi

Metoda	Povprečen uspeh pri iskanju				Povprečno št. iteracij				M. dela
Navadna iteracijska	0.30%	0.35%	0.70%	**0.45%**	40.5	444.6	326.9	**271***	**60143***
Bisekcijska	35.8%	38.8%	39.5%	**38.0%**	13.4	13.2	13.6	**13.4**	**35.2**
Sekantna	95.6%	91.6%	94.8%	**94.0%**	8.7	8.8	9.1	**8.9**	**9.4**
Newtonova	99.9%	99.6%	100.0%	**99.8%**	7.0	7.2	7.2	**7.1**	**7.1**
Kvadr. Newtonova	29.5%	32.5%	31.1%	**31.0%**	2.4	2.5	2.6	**2.5**	**8.1**

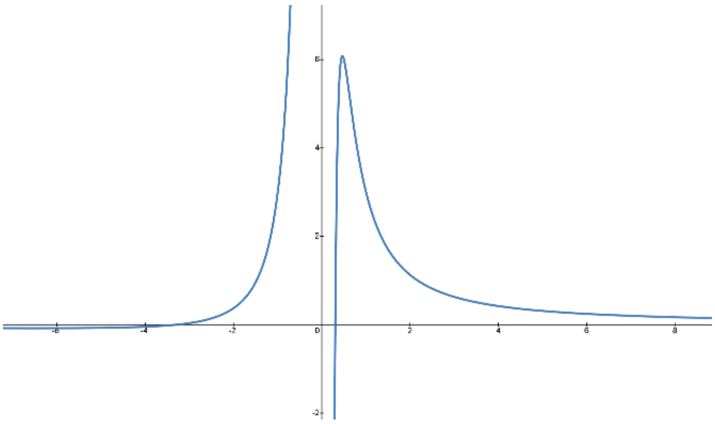
*Nezanesljive vrednosti zaradi majhnega vzorca (malo število primernih enačb).

Po pričakovanjih je bila Newtonova metoda najuspešnejša, saj skoraj ni bilo neuspešnih poskusov (razen nekaterih lokalnih minimumov in maksimumov, ki so ujeli algoritem), navadna iteracijska metoda pa je delovala le v zelo redkih primerih. Sekantna metoda je bila hitrejša in uspešnejša kot bisekcijska. Bisekcija ima nizko stopnjo uspešnosti, ker so ugibanja pogosto na isti strani vodoravne osi, pri čemer je neuporabna. Kvadratna Newtonova metoda je bila hitrejša od običajne Newtonove metode, vendar je delovala v približno vsakem tretjem primeru. Poleg tega se Newtonovo metodo lahko hitreje uporablja z navadnim znanstvenim kalkulatorjem, saj jo je mogoče ponoviti s funkcijo *ANS*, kar pa ne velja za kvadratno Newtonovo metodo.

Racionalne funkcije (2-5 členov; stopnje (-5)-0; koeficienti (-5)-5; natančnost 3; 20 začetnih približkov/enačbo; 3 x 100 enačb)

Tabela 5: Zmogljivost metod: racionalne funkcije

Metoda	Povprečen uspeh pri iskanju				Povprečno št. iteracij				M. dela
Navadna iteracijska	76.7%	68.9%	63.9%	**69.9%**	7.7	6.7	9.0	**7.8**	**11.2**
Bisekcijska	17.6%	6.7%	21.5%	**15.3%**	11.8	11.2	11.7	**11.6**	**75.7**
Sekantna	19.7%	17.5%	17.5%	**18.2%**	7.0	6.3	7.7	**7.0**	**38.6**
Newtonova	32.3%	23.6%	33.7%	**29.9%**	5.8	5.6	8.7	**6.7**	**22.4**
Kvadr. Newtonova	16.6%	15.7%	12.6%	**15.0%**	3.1	2.9	3.3	**3.1**	**20.6**



Graf 6: Primer grafa funkcije iz kategorije racionalnih funkcij

Newtonova metoda pri tej vrsti enačb ne deluje najbolje, saj obstajajo vodoravne asimptote, zaradi katerih se približki oddaljijo, če začetni približek ni na pravem mestu. Podobne težave ima tudi sekantna metoda. Bisekcija deluje še slabše kot pri preprostih polinomih, saj je vodoravna asimptota pogosto bližje osi x v neskončnost in izbira napačno točko kot naslednji približek. Kvadratna Newtonova metoda deluje najhitreje, vendar ima nizko stopnjo uspešnosti. Po drugi strani je navadna iteracijska metoda pri takih enačbah najuspešnejša in zelo hitra, poleg tega pa je njeno računanje preprosto.

Kombinirane funkcije (polinomski + racionalni členi) (2-5 členov; stopnje (-5)-5; koeficienti (-5)-5; natančnost 3; 20 začetnih približkov/enačbo; 3 x 100 enačb)

Tabela 6: Zmogljivost metod: kombinirane funkcije

Metoda	Povprečen uspeh pri iskanju				Povprečno št. iteracij				M. dela
Navadna iteracijska	17.7%	15.5%	15.1%	**16.1%**	6.4	10.7	4.7	**7.3**	**45.2**
Bisekcijska	27.1%	22.8%	18.7%	**22.8%**	11.7	11.9	12.5	**12.0**	**52.7**
Sekantna	64.0%	72.2%	63.4%	**66.5%**	9.8	9.6	11.0	**10.1**	**15.3**
Newtonova	74.3%	83.3%	75.5%	**77.7%**	11.7	9.3	10.5	**10.5**	**13.5**
Kvadr. Newtonova	23.3%	22.7%	23.2%	**23.1%**	2.7	2.7	2.8	**2.7**	**11.9**

Enačbe iz te kategorije večinoma nimajo vodoravnih asimptot, zato je Newtonova metoda zanje uporabna. Kvadratna Newtonova metoda je najhitrejša in ima nižje merilo dela kot Newtonova metoda, vendar je ni mogoče preprosto ponoviti z *ANS* na kalkulatorju in zahteva več korakov, zaradi česar je za to vrsto enačb najboljša Newtonova metoda. Tudi sekantna metoda je tu precej uporabna, vendar zahteva delo z dvema spremenljivkama (približka), zaradi česar je časovno zahtevnejša od Newtonove metode. Navadna iteracijska metoda se večinoma najbolje obnese pri funkcijah, ki jih Newtonova metoda ne more rešiti, če vodilni člen ni stopnje, večje od 1 in ima člene negativnih stopenj.

Zaključek

V tej raziskovalni nalogi sem raziskal učinkovitost različnih numeričnih metod za aproksimacijo ničel funkcij pri reševanju z uporabo papirja, pisala in navadnega znanstvenega kalkulatorja. Nekatere bolj znane metode sem primerjal na različnih vrstah enačb. Navadna iteracijska metoda, ki je neuporabna za navadne polinome, prekaša vse druge metode, kadar so stopnje vseh členov enake ali manjše od 0, za druge vrste pa je najboljša Newtonova metoda - najhitrejša in dovolj preprosta (če je funkcija odvedljiva). Sekantna metoda je primerljiva, vendar nekoliko počasnejša in nekoliko manj uspešna. Bisekcija zahteva 2 primerna začetna približka, zaradi česar je njena ocena slaba, saj ugibana začetnih približkov pogosto niso ustrezna. Na primer preprosta konkavna funkcija z dvema ničlama pri -1 in +1 bo velikokrat neuspešna, saj so začetni približki naključno generirani v razponu od -5 do 5, tako pa večina kombinacij popolnoma izpade iz tega razpona in algoritem zabeleži neuspešen poskus. Na novo zasnovana metoda (kvadratna Newtonova) je veliko hitrejša (glede na potrebne iteracije za želeno natančnost) od drugih, vendar zahteva veliko več korakov in ima bistveno nižjo stopnjo uspešnosti. Zaključimo lahko, da je Newtonova metoda še vedno najboljši način za aproksimacijo ničel polinomov, navadna iteracijska metoda pa je presenetljivo (po rezultatih iz prvega niza polinomov) najboljša za številne racionalne funkcije. Je tudi najpreprostejša.

Prednost te raziskave je bila uporaba računalniškega algoritma za večkratno izvajanje vseh algoritmov na več različnih enačbah in iz različnih začetnih približkov, kar bi za izračun s človeško roko verjetno trajalo več let ali celo desetletij. Obdelava podatkov je na računalniku tudi preprostejša, rezultati pa niso izpostavljeni pristranskosti raziskovalca, saj so bili algoritmi napisani pred primerjavo uspešnosti in so bili testirani ločeno z različnimi začetnimi nastavitvami, zato ni mogoče napisati enega algoritma, ki bi bil uspešnejši od drugih in bi potrdil raziskovalčevo hipotezo.

Vendar pa obstaja nekaj omejitev in možnih izboljšav te raziskave, prva in najbolj očitna je oblikovanje algoritma, ki bi sprejemal vse osnovne funkcije - trigonometrične, logaritemske, eksponentne, hiperbolične ... Druga možna izboljšava za res "pošteno" primerjavo metod bi bil dober algoritem za preoblikovanje enačbe za navadno iteracijsko metodo, da bi iskal iste ničle kot vse druge metode namesto rešitev $x = f(x)$, kot je to delal v tej raziskavi. Toda, kot sem že omenil, vedno obstajajo različni načini za preoblikovanje enačbe v ustrezno obliko in vsi niso enako uspešni pri uporabi metode navadne iteracije. S tem se poveča tudi število potrebnih korakov in spremeni se učinkovitost ročnega računanja. V nadaljnjih raziskavah bi lahko raziskali še en vidik različnih metod za aproksimacijo – učinkovitost metod na računalnikih (in kalkulatorjih), saj operacije za aproksimacijo rešitev potrebujejo različen čas in količino pomnilnika. Obstaja tudi več drugih metod, ki so za človeka precej zamudne, vendar so na računalniku veliko hitrejše od mnogih, opisanih v tej raziskovalni nalogi.

Literatura

He, T.-X., & Shiue, P. J.-S. (2012). A Note on Horner's Method. Dostop: https://www.researchgate.net/profile/Tian-Xiao-He-2/publication/268615911_A_note_on_Horner's_method/links/57c0865508aed246b0fb61b1/A-note-on-Horners-method.pdf (19. 8. 2022)

Yew, A. C. (2011). Fixed point iterations. Dostop: https://www.dam.brown.edu/people/alcyew/handouts/fixedpoint_iterations.pdf (18. 8. 2022)