**math.e**

*Hrvatski matematički elektronički časopis*

# Približno računanje kvadratnog korijena

**Niko Josipović**
Učenik III. razreda Tehničke škole Ruđera Boškovića

**Marijana Krnić**
Profesorica matematike, izvrstan savjetnik u Tehničkoj školi Ruđera Boškovića

## 1 UVOD

U matematici, kao i u mnogim drugim znanostima, pored osnovnih aritmetičkih operacija zbrajanja, oduzimanja, množenja i dijeljenja, često se susrećemo s operacijama potenciranja i korjenovanja. Primjerice, izračunavanje površine kvadrata s poznatom duljinom stranice zahtijeva kvadriranje te duljine. Međutim postavlja se pitanje: kolika je duljina stranice kvadrata ako je poznata njegova površina?

Ovakva vrsta pitanja uvodi nas u koncept kvadratnog korijena. No, kako uopće izračunati drugi korijen iz broja koji nije potpun kvadrat kao što su četiri ili devet?

U ovom radu ćemo prikazati metode korištene za približno računanje kvadratnog korijena, Babilonsku i Newtonovu metodu. Razmotrit ćemo i heuristički pristup problemu iz kojeg proizlazi nekoliko zanimljivih rezultata. Uz to, u radu su dane implementacije pojedinih postupaka u programskim jezicima C i Python.

## 2 BABILONSKA METODA

Jedna od najranijih poznatih metoda za približno računanje kvadratnog korijena potječe iz vremena starog Babilona, zbog čega je poznata kao Babilonska metoda [4]. Riječ je o metodi koja za početnu vrijednost drugog korijena pozitivnog realnog broja $a$ uzima neku pozitivnu vrijednost $x_0$, da bi se sljedeća, bolja aproksimacija $x_1$ izračunavala kao aritmetička sredina brojeva $x_0$ i $\frac{a}{x_0}$:

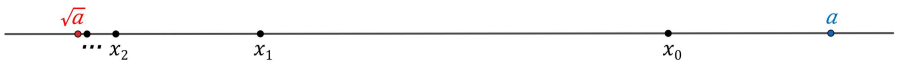
$$x_1 = \frac{1}{2}\left(x_0 + \frac{a}{x_0}\right).$$

Postupak nastavljamo induktivno pa formula općenito glasi:

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right), \tag{1}$$

gdje $n$ označava redni broj iteracije. Kako $n$ teži u beskonačnost, vrijednost $x_n$ se približava vrijednosti kvadratnog korijena broja $a$:

$$\lim_{n\to\infty} x_n = \sqrt{a}.$$




Slika 1: *Prikaz približavanja aproksimacija Babilonske formule (1) kvadratnom korijenu*

### 2.1 NUMERIČKI PRIMJER

Svojstva Babilonske metode pokazat ćemo kroz sljedeći primjer. Pretpostavimo da tražimo vrijednost kvadratnog korijena broja dva, odnosno neka je $a = 2$. Započinjemo s početnom vrijednošću $x_0 = 5$. Koristeći (1), dobivamo niz aproksimacija prikazanih u sljedećoj tablici (uz prikaz greške aproksimacija).

Tablica 1: *Vrijednosti aproksimacija Babilonske formule*

| $n$ | $x_n$ | $\lvert x_n - \sqrt{2}\rvert$ |
|---|---|---|
| 0 | $5$ | $3.586 \cdot 10^{0}$ |
| 1 | $2.7$ | $1.286 \cdot 10^{0}$ |
| 2 | $1.72037$ | $3.062 \cdot 10^{-1}$ |
| 3 | $1.414215686275\ldots$ | $2.724 \cdot 10^{-2}$ |
| 4 | $1.414213562375\ldots$ | $2.574 \cdot 10^{-4}$ |
| 5 | $1.414213562373\ldots$ | $2.342 \cdot 10^{-8}$ |

Uočavamo da je svaka sljedeća aproksimacija preciznija od prethodne. Također, promatrajući vrijednosti grešaka, primjećujemo kvadratnu konvergenciju, jer se svaka sljedeća greška smanjuje barem kvadratno u odnosu na prethodnu.
Prvo intuitivno objašnjenje učinkovitosti metode dao je Isaac Newton (1643. - 1727.), pokazujući da je to poseban slučaj Newtonove metode [3].
*Program.* Implementacija postupka približnog računanja kvadratnog korijena korištenjem Babilonske formule u programskom jeziku C može se pronaći u poglavlju *Programi* pod brojem 1.

# 3 NEWTONOVA METODA

Newtonova metoda ili *metoda tangente* predstavlja iterativni postupak koji se koristi za približno rješavanje nelinearnih jednadžbi. Postupak započinjemo početnom procjenom rješenja, a zatim se uzastopnim iteracijama postupno poboljšava. Njezina popularnost proizlazi iz učinkovitosti rješavanja jednadžbi za koje nemamo izravna rješenja, te brzine konvergencije [5].

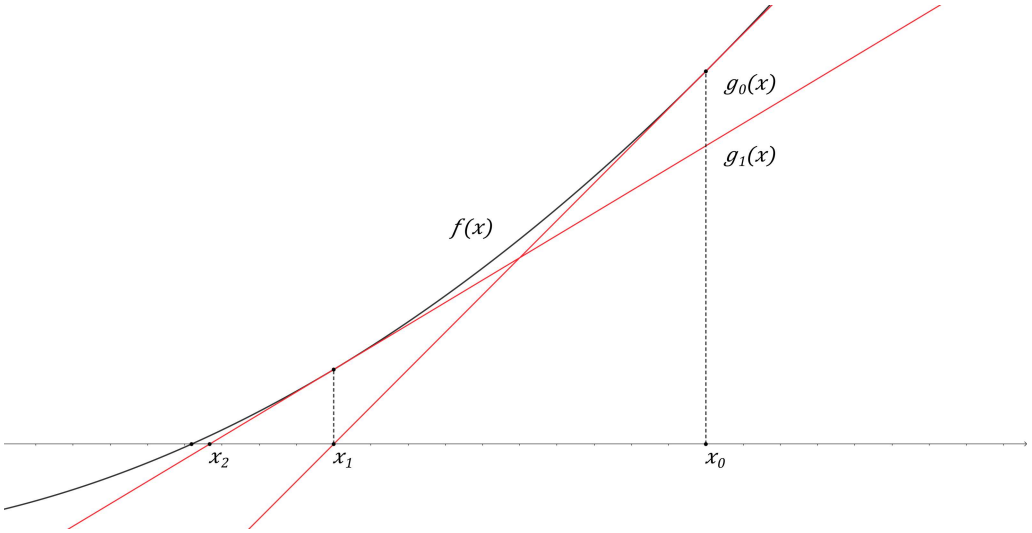
## 3.1 IDEJA NEWTONOVE METODE I NJEN GEOMETRIJSKI IZVOD

Ideja Newtonove metode leži u upotrebi tangente kako bi aproksimirali nultočku neke funkcije [5]. Neka je $f$ diferencijabilna funkcija definirana na intervalu $I$ unutar kojeg postoji jedinstvena nultočka. Uz početnu pretpostavku $x_0$ o rješenju, možemo odrediti tangentu na graf funkcije $f$ u točki $T_0(x_0, f(x_0))$. Jednadžba te tangente je $g(x) = f(x_0) + f'(x_0)(x - x_0)$, a ideja je pronaći njenu nultočku $x_1$, uz uvjet da ona postoji. Uvjet da $g$ ima nultočku je $f'(x_0) \neq 0$ i $f(x_0) \neq 0$, pri čemu drugi uvjet sugerira da smo već pronašli vrijednost funkcije koju tražimo. Ako u jednadžbu za tangentu (linearnu aproksimaciju) uvrstimo činjenicu da je $g(x) = 0$, izraz za $x_1$ glasi:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Ponavljajući induktivno ovaj postupak za svaku sljedeću aproksimaciju nultočke funkcije $f$ dobivamo:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n = 0, 1, 2, \ldots \tag{2}$$




Slika 2: *Grafički prikaz Newtonove metode*

## 3.2 PRIMJENA NEWTONOVE METODE

Neka je zadan pozitivan realan broj $a$. Cilj je pronaći $x$, takav da vrijedi $x = \pm\sqrt{a}$, odnosno $x^2 = a$. Definirajmo funkciju $f(x) = x^2 - a$. Primjećujemo da je problem određivanja kvadratnog korijena sveden na traženje nultočaka te funkcije. Budući da je ova funkcija diferencijabilna i ima jedinstveno pozitivno i negativno rješenje (nultočke su izolirane), možemo primijeniti Newtonovu metodu. Uvrštavanjem $f(x) = x^2 - a$, te $f'(x) = 2 \cdot x$ u (2), izvodimo formulu ekvivalentnu Babilonskoj (1):

$$\begin{aligned} x_{n+1} &= x_n - \frac{f(x_n)}{f'(x_n)} \\ &= x_n - \frac{x_n^2 - a}{2 \cdot x_n} \\ &= \frac{2 \cdot x_n^2}{2 \cdot x_n} - \frac{x_n^2 - a}{2 \cdot x_n} \\ &= \frac{x_n^2 + a}{2 \cdot x_n} \\ x_{n+1} &= \frac{1}{2}\left(x_n + \frac{a}{x_n}\right). \end{aligned}$$

# 4 HEURISTIČKI PRISTUP

Primjenom Newtonove metode došli smo do Babilonske formule i objašnjenja njene učinkovitosti. Međutim, za ovo objašnjenje potrebno je poznavanje infinitezimalnog računa, što Babiloncima nije bilo poznato [3]. Stoga ćemo do formule i objašnjenja doći na jednostavniji način.

Dan je izraz $x^2 = a$. Transformacijom jednadžbe, izvodimo izraze za $x$:

Tablica 2: *Prikaz izvođenja izraza za $x$*

| $x^2 = a$ | | |
|---|---|---|
| $x^2 + x = a + x$ | $x^2 + 2x = a + 2x$ | $x^2 + 3x = a + 3x$ |
| [-5pt] $x\,(x+1) = a + x$ | $x\,(x+2) = a + 2x$ | $x\,(x+3) = a + 3x$ |
| $x = \dfrac{a+x}{x+1}$ | $x = \dfrac{a+2x}{x+2}$ | $x = \dfrac{a+3x}{x+3}$ |

Dakle, općenito pišemo:

$$x = \frac{a + k \cdot x}{x + k}, k \in \mathbb{R}, x + k \neq 0. \tag{3}$$

Izveden općeniti izraz za $x$, (3), pripada klasi jednostavnih iteracija, jer je oblika

$$x = \varphi(x).$$

Tako smo problem rješavanja početne jednadžbe sveli na problem traženja fiksne točke[1] funkcije $\varphi(x)$. Da bismo ju pronašli, definiramo jednostavnu iteracijsku funkciju (4) za izraz (3):

$$x_{n+1} = \varphi\left(x_n\right), n = 0, 1, 2, \ldots \tag{4}$$

$$x_{n+1} = \frac{a + k \cdot x_n}{x_n + k}. \tag{5}$$

Navedena iteracijska funkcija (5) može se, a i ne mora, približavati vrijednosti fiksne točke $x$, kako $n$ teži k beskonačnosti, odnosno u ovom slučaju kvadratnom korijenu broja $a$. U svrhu pronalaženja izraza koji konvergiraju koristimo sljedeće teoreme:

**Teorem 1.** (Teorem o fiksnoj točki) *Neka je* $\varphi \in C^1\left[a, b\right]$ za koju vrijedi:
1. $\varphi\left(\left[a, b\right]\right) \subseteq \left[a, b\right]$
2. i neka $\exists \lambda \in \left[0, 1\right\rangle$, takav da je $\left|\varphi'(x)\right| \leq \lambda, \forall x \in \left[a, b\right]$.
Tada $x = \varphi(x)$ ima točno jedno rješenje $\alpha$ na $\left[a, b\right]$, te za proizvoljan $x_0 \in \left[a, b\right]$ i jednostavnu iteraciju $x_{n+1} = \varphi(x_n), n \geq 0$ vrijedi

$$\lim_{n \to \infty} x_n = \alpha.$$

**Teorem 2.** *Neka je* $\alpha$ rješenje jednostavne iteracije $x = \varphi(x)$, $\varphi$ neprekidno diferencijabilna na nekoj okolini od $\alpha$ i $\left|\varphi'(\alpha)\right| < 1$. Tada vrijede svi rezultati Teorema 1, uz pretpostavku da je $x_0$ dovoljno blizu $\alpha$.
Dokazi navedenih teorema mogu se pronaći u [1].

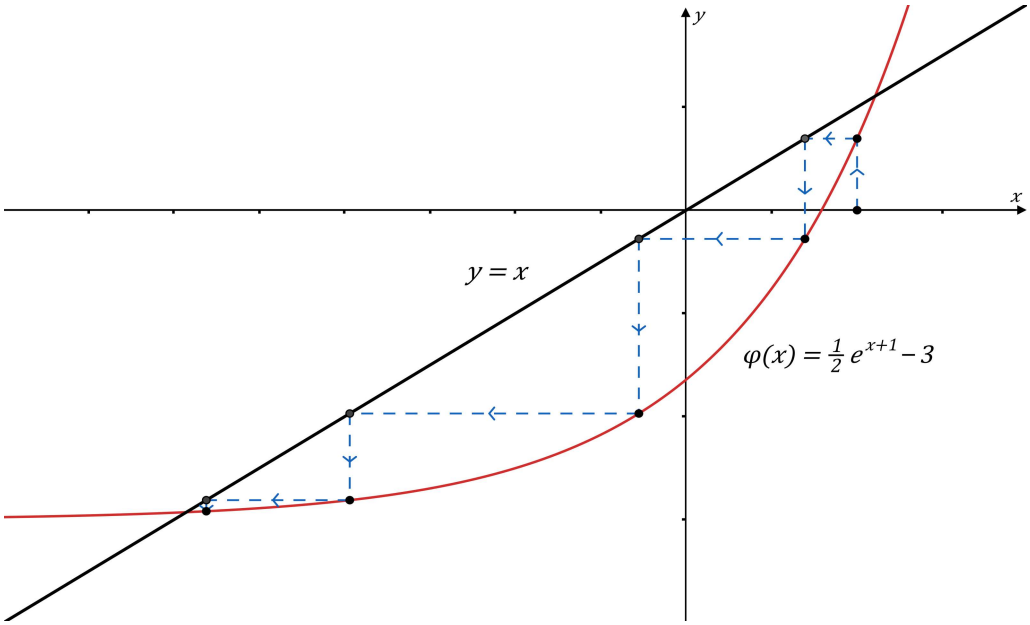
Kako je $\varphi(x) = \dfrac{a + k \cdot x}{x + k}$, njezina je derivacija $\varphi'(x) = \dfrac{k^2 - a}{(x+k)^2}$. Također, poznata nam je jednakost $\alpha = \sqrt{a}$, pa prema Teoremu 2 gledamo kada vrijedi sljedeće:
[-15pt]

$$\left|\varphi'(\alpha)\right| < 1,$$

odnosno

$$-1 < \frac{k^2 - a}{\left(\sqrt{a} + k\right)^2} < 1.$$

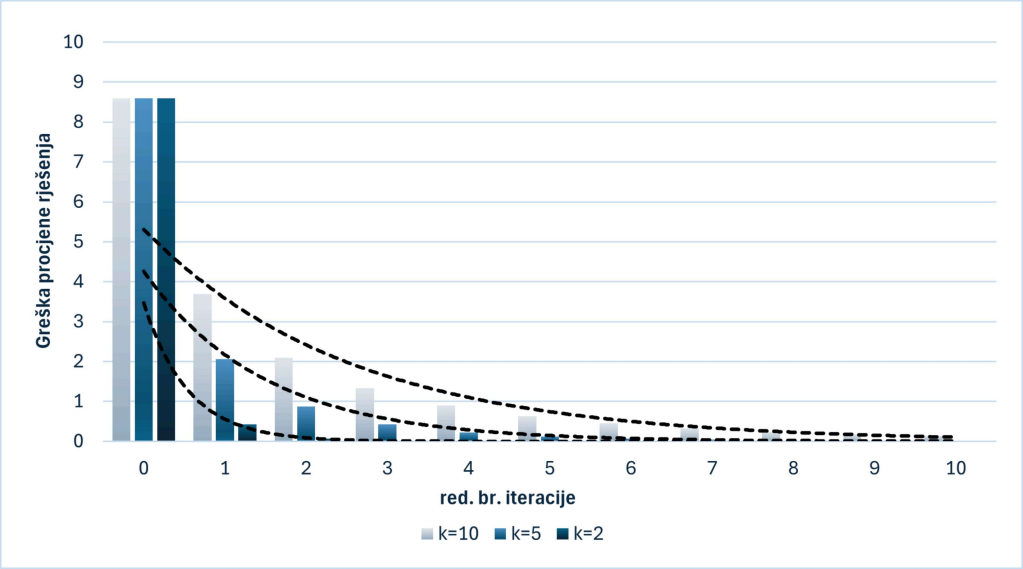
Rješavanjem nejednadžbe, dolazimo do uvjeta koeficijenta $k > 0$, što znači da, ako iteracijska funkcija (5) zadovoljava taj uvjet, ona uspješno konvergira u okolini $\sqrt{a}$.




Slika 3: *Geometrijski prikaz približavanja iteracijske funkcije,* $x_{n+1} = \frac{1}{2}e^{x_n+1} - 3$, fiksnoj točci

## 4.1 UTJECAJ KOEFICIJENTA $k$

Primjena opće iteracijske funkcije (5) za približno računanje kvadratnog korijena broja $a = 2$ pri različitim vrijednostima $k > 0$, dovodi do zanimljivog rezultata prikazanog na grafikonu 1.




Grafikon 1: *Prikaz greške aproksimacija formula različitih vrijednosti* $k > 0$

Uočavamo da pri različitim vrijednostima koeficijenta $k$ dobivamo formule s različitim brzinama konvergencije. Pritom se nameće pitanje: koja vrijednost $k$ daje najbržu konvergenciju?

S ciljem pronalaženja takvog $k$ razmatramo slučaj, gdje je rezultat prve iteracije već konačan, odnosno $x_1 = \sqrt{a}$, za bilo koju početnu vrijednost $x_0$. Uvrštavanjem u (5) dobivamo:

$$\sqrt{a} = \frac{a + k \cdot x_0}{x_0 + k}.$$

Izrazimo $k$ iz prethodne formule:

$$\sqrt{a} \cdot x_0 + \sqrt{a} \cdot k = x_0 \cdot k + a$$

$$k(\sqrt{a} - x_0) = a - \sqrt{a} \cdot x_0$$

$$k = \frac{a - \sqrt{a} \cdot x_0}{\sqrt{a} - x_0}$$

$$k = \frac{\sqrt{a}(\sqrt{a} - x_0)}{\sqrt{a} - x_0}$$

$$k = \sqrt{a}.$$

Dobivamo da je za $k = \sqrt{a}$ konvergencija najbrža. Stoga, izjednačimo $k$ s aproksimacijom $\sqrt{a}$, odnosno s $x$. Uvrštavanjem u (5) izvodimo „dinamičku" formulu (6), ekvivalentnu Babilonskoj. Dinamičku, jer je svaka sljedeća, bolja aproksimacija istovremeno nova vrijednost za $k$ u idućoj iteraciji, čime uvjetujemo vrlo brzu konvergenciju formule

$$x_{n+1} = \frac{a + x_n^2}{2 \cdot x_n}. \tag{6}$$

*Program.* Implementacija izračunavanja greške procjene rješenja pri različitim vrijednostima $k$ i grafički prikaz tih rezultata u programskom jeziku Python prikazana je u poglavlju *Programi* pod br. 2.

## 4.2 FORMULE ŽELJENOG REDA KONVERGENCIJE

Izjednačavanje $k$ s $x$ iznjedrilo je Babilonsku formulu, formulu vrlo brze konvergencije. To nas dovodi do idućeg pitanja: što ako za $k$ uvrstimo bolju aproksimaciju od $x$, odnosno $k$ izjednačimo s prethodno dobivenom formulom (6)

$$k = \frac{a + x^2}{2 \cdot x}?$$

Uvrštavanjem u (5) dolazimo do nove, brže formule za približno računanje drugog korijena.

$$x_{n+1} = \frac{a + \dfrac{a + x_n^2}{2 \cdot x_n} \cdot x_n}{x_n + \dfrac{a + x_n^2}{2 \cdot x_n}}$$

$$= \frac{x_n \left(2 \cdot a + a + x_n^2\right)}{2 \cdot x_n^2 + a + x_n^2}$$

$$x_{n+1} = \frac{x_n^3 + 3 \cdot x_n \cdot a}{3 \cdot x_n^2 + a}. \tag{7}$$

Postupak izjednačavanja $k$ s novodobivenom formulom (isprva s $x$) i uvrštavanje u (5) rezultira sljedećim formulama:

Tablica 3: *Prve četiri formule dobivene ponavljanjem opisanog postupka*

| red konvergencije | popis formula |
|---|---|
| 2 | $x_{n+1} = \frac{x_n^2 + a}{2x_n}$ |
| 3 | $x_{n+1} = \frac{x_n^3 + 3x_n a}{3x_n^2 + a}$ |
| 4 | $x_{n+1} = \frac{x_n^4 + 6x_n^2 a + a^2}{4x_n^3 + 4x_n a}$ |
| 5 | $x_{n+1} = \frac{x_n^5 + 10x_n^3 a + 5x_n a^2}{5x_n^4 + 10x_n^2 a + a^2}$ |

Zanimljivo je kako formule imaju za jedan veći red konvergencije[2] od prethodne. Kako bismo provjerili red konvergencije pojedine formule, koristimo Teorem 3.
**Teorem 3.** *Neka je* $\alpha$ rješenje od $x = \varphi(x)$ i neka je $\varphi p$ puta neprekidno diferencijabilna za sve $x$ u okolini $\alpha$, za neki $p \geq 2$. Nadalje, pretpostavimo da je

$$\varphi'(\alpha) = \cdots = \varphi^{(p-1)}(\alpha) = 0. \tag{8}$$

Ako je početna vrijednost $x_0$ dovoljno blizu $\alpha$, iteracijska funkcija

$$x_{n+1} = \varphi(x_n), n \geq 0$$

imat će red konvergencije $p$.

U literaturi je prikazan i dokaz ovog teorema [1].

Tako primjerice, formule (6) i (7) korištenjem Teorema 3, u kojem razmatramo do kojeg $p$ vrijedi niz jednakosti (8), imaju red konvergencije 2 i 3. Dolazak do toga jasno je prikazan u tablici 4.

Tablica 4: *Postupak pronalaženja reda konvergencije formula (6) i (7)*

| $\varphi(x)$ | $\frac{x^2 + a}{2x}$ | $\frac{x^3 + 3xa}{3x^2 + a}$ |
|---|---|---|
| $\varphi'(\sqrt{a})$ | $0$ | $0$ |
| $\varphi''(\sqrt{a})$ | $\frac{1}{\sqrt{a}}$ | $0$ |
| $\varphi'''(\sqrt{a})$ | | $\frac{-3}{2a}$ |
| $p$ | $2$ | $3$ |

Na temelju navedenog, postavljamo sljedeću hipotezu:

**Hipoteza 1.** *Ponavljanjem opisanog postupka* $n$ puta dobiva se formula reda konvergencije $n + 1$.

Da bismo dokazali istinitost hipoteze, analizirajmo formule dobivene opisanim postupkom, navedene u tablici 3. Uočavamo da su članovi brojnika svi neparni članovi razvoja $(x_n+\sqrt{a})^{m+1}$ u binomni red, što se može zapisat u obliku:

$$\frac{(x_n+\sqrt{a})^{m+1}+(x_n-\sqrt{a})^{m+1}}{2}. \tag{9}$$

Slično tome, nazivnik čine svi parni članovi istog razvoja podijeljeni s $\sqrt{a}$, što može biti zapisano kao:

$$\frac{(x_n+\sqrt{a})^{m+1}-(x_n-\sqrt{a})^{m+1}}{2\cdot\sqrt{a}}. \tag{10}$$

Iako su pravilnosti potvrđene samo za $m \leq 4$, gdje je $m \in \mathbb{N}$, pretpostavit ćemo da one vrijede općenito, tj. za $\forall m \in \mathbb{N}$. Konačno, postupak svodimo na sljedeći izraz:

$$x_{n+1}=\frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{m+1}+(x_n-\sqrt{a})^{m+1}\right]}{(x_n+\sqrt{a})^{m+1}-(x_n-\sqrt{a})^{m+1}}, \tag{11}$$

gdje izborom $m$ konstruiramo formulu koju bismo dobili ponavljanjem postupka $m$ puta.

Pretpostavka o općenitosti uočenih pravilnosti ne mora biti istinita. Stoga je potrebno dokazati da izraz (11) zaista konstruira formule na opisan način.

*Dokaz.* Dokaz ćemo provesti indukcijom.

*Baza indukcije.* Provjerimo da izraz (11) konstruira ispravnu formulu za $m=1$. Uvrštavanjem $m=1$ u (11) i sređivanjem dobivamo:

$$\begin{aligned}
x_{n+1} &= \frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^2+(x_n-\sqrt{a})^2\right]}{(x_n+\sqrt{a})^2-(x_n-\sqrt{a})^2} \\
&= \frac{\sqrt{a}\cdot\left(x_n^2+2\cdot x_n\cdot\sqrt{a}+a+x_n^2-2\cdot x_n\cdot\sqrt{a}+a\right)}{(x_n+\sqrt{a}-x_n+\sqrt{a})\,(x_n+\sqrt{a}+x_n-\sqrt{a})} \\
&= \frac{\sqrt{a}\cdot(2\cdot x_n^2+2\cdot a)}{4\cdot x_n\cdot\sqrt{a}} \\
&= \frac{x_n^2+a}{2\cdot x_n}
\end{aligned}$$

formulu ekvivalentnoj (6), što znači da je tvrdnja istinita za $m=1$. *Pretpostavka indukcije.* Pretpostavimo da za neki $s \in \mathbb{N}$ vrijedi:

$$x_{n+1}=\frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}+(x_n-\sqrt{a})^{s+1}\right]}{(x_n+\sqrt{a})^{s+1}-(x_n-\sqrt{a})^{s+1}}.$$

*Korak indukcije.* Pokažimo koristeći pretpostavku da vrijedi sljedeća jednakost:

$$\frac{a+k\cdot x_n}{x_n+k}=\frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+2}+(x_n-\sqrt{a})^{s+2}\right]}{(x_n+\sqrt{a})^{s+2}-(x_n-\sqrt{a})^{s+2}}. \tag{12}$$

$$\frac{a+k\cdot x_n}{x_n+k}=\frac{a+\dfrac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}+(x_n-\sqrt{a})^{s+1}\right]}{(x_n+\sqrt{a})^{s+1}-(x_n-\sqrt{a})^{s+1}}\cdot x_n}{x_n+\dfrac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}+(x_n-\sqrt{a})^{s+1}\right]}{(x_n+\sqrt{a})^{s+1}-(x_n-\sqrt{a})^{s+1}}}$$

$$=\frac{\sqrt{a}\cdot\left[\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}-(x_n-\sqrt{a})^{s+1}\right]+\left[(x_n+\sqrt{a})^{s+1}+(x_n-\sqrt{a})^{s+1}\right]\cdot x_n\right]}{x_n\cdot\left[(x_n+\sqrt{a})^{s+1}-(x_n-\sqrt{a})^{s+1}\right]+\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}+(x_n-\sqrt{a})^{s+1}\right]}$$

$$=\frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+1}\cdot(x_n+\sqrt{a})+(x_n-\sqrt{a})^{s+1}\cdot(x_n-\sqrt{a})\right]}{(x_n+\sqrt{a})^{s+1}\cdot(x_n+\sqrt{a})-(x_n-\sqrt{a})^{s+1}\cdot(x_n-\sqrt{a})}$$

$$=\frac{\sqrt{a}\cdot\left[(x_n+\sqrt{a})^{s+2}+(x_n-\sqrt{a})^{s+2}\right]}{(x_n+\sqrt{a})^{s+2}-(x_n-\sqrt{a})^{s+2}}.$$

Dobili smo jednakost (12), pa smo po principu potpune indukcije dokazali tvrdnju da izraz (11) konstruira formulu, za $\forall m\in\mathbb{N}$, koju bismo dobili ponavljanjem postupka $m$ puta.

Izjednačimo izraz (11) s $\varphi(x)$ i izračunajmo prvu derivaciju funkcije $\varphi$.

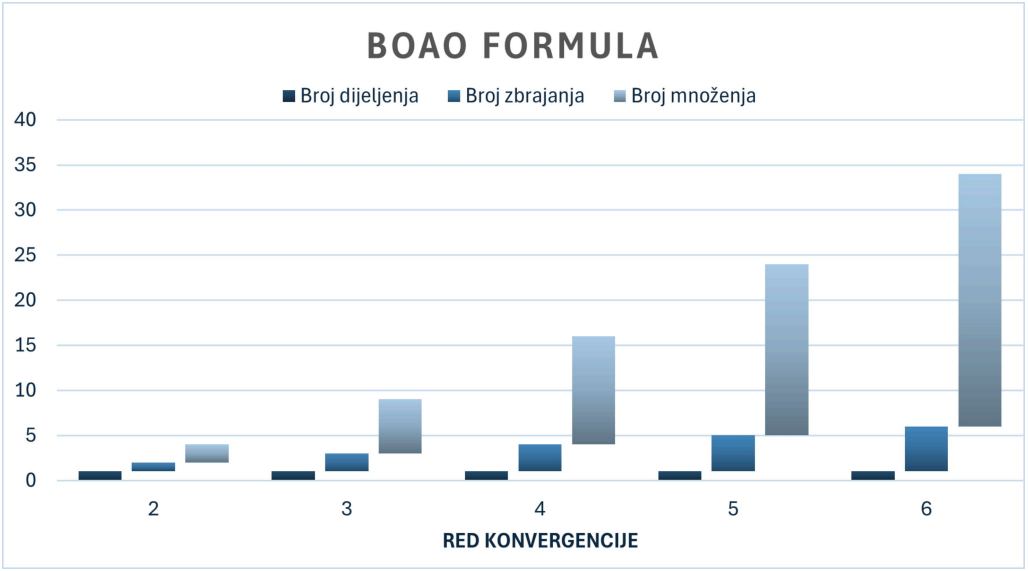
$$\varphi(x)=\frac{\sqrt{a}\cdot\left[(x+\sqrt{a})^{m+1}+(x-\sqrt{a})^{m+1}\right]}{(x+\sqrt{a})^{m+1}-(x-\sqrt{a})^{m+1}}$$

$$\varphi'(x)=\frac{4\cdot a\cdot(m+1)\cdot(x+\sqrt{a})^{m}\cdot(x-\sqrt{a})^{m}}{\left[(x+\sqrt{a})^{m+1}-(x-\sqrt{a})^{m+1}\right]^2}$$

Funkcija $\varphi(x)$ je glatka[3], a njena prva derivacija u brojniku sadrži izraz $(x-\sqrt{a})^m$. Dakle, niz jednakosti u (8) će zbog svojstva derivacije kompozicije i kvocijenta funkcije vrijediti zaključno s $p=m+1$. Time je Hipoteza 1 dokazana.

## 4.3 ANALIZA FORMULA

Na temelju prethodnog, moguće je konstruirati formule visokog reda konvergencije. Ipak, zbog velikog broja osnovnih aritmetičkih operacija (BOAO) po iteraciji, što je prikazano na grafikonu 2, takve se formule ne koriste.




Grafikon 2: *Prikaz broja osnovnih aritmetičkih operacija formula, bez minimizacija*

Kako bismo olakšali proces određivanja BOAO pojedine formule, implementirat ćemo ga u programski jezik C. Za to je potrebno izraz (11) pretvoriti u sumu koristeći binomnu formulu, što ćemo učiniti posebno za brojnik (9) i nazivnik (10):

$$\frac{(x_n+\sqrt{a})^{m+1}+(x_n-\sqrt{a})^{m+1}}{2}=\sum_{i=0}\binom{m+1}{2i}\cdot x_n^{m+1-2i}\cdot a^i$$

$$\frac{(x_n+\sqrt{a})^{m+1}-(x_n-\sqrt{a})^{m+1}}{2\cdot\sqrt{a}}=\sum_{i=0}\binom{m+1}{2i+1}\cdot x_n^{m-2i}\cdot a^i.$$

Uvrštavanjem dobivamo izvedenicu izraza (11) koja glasi:

$$x_{n+1}=\frac{\displaystyle\sum_{i=0}\binom{m+1}{2i}\cdot x_n^{m+1-2i}\cdot a^i}{\displaystyle\sum_{i=0}\binom{m+1}{2i+1}\cdot x_n^{m-2i}\cdot a^i}. \tag{13}$$

Spomenuti program nalazi se u poglavlju *Programi* pod br. 3, a prilikom korištenja uočavamo pravilnost prikazanu u tablici 5.

Tablica 5: *Pravilnost u nizu ukupnog broja osnovnih aritmetičkih operacija formula*

| $m$ | br. operacija | promjena | br. operacija | promjena promjene |
|---|---|---|---|---|
| 1 | 4 | +4 | | |
| 2 | 9 | +5 | | +1 |
| 3 | 16 | +7 | | +2 |
| 4 | 24 | +8 | | +1 |
| 5 | 34 | +10 | | +2 |
| 6 | 45 | +11 | | +1 |
| 7 | 58 | +13 | | +2 |

Uočavamo da niz vrijednosti promjena promjene oscilira, tako da za parne $m$ iznosi $1$, a neparne iznosi $2$ (isključeno $m=1$). Pod pretpostavkom općenitosti, dokažimo da broj operacija za neki $m\in\mathbb{N}$ u oznaci $S(m)$ opisujemo formulom:

$$S(m)=4\cdot m+3\cdot\left\lfloor\frac{m^2}{4}\right\rfloor-2\cdot\left\lfloor\frac{m}{2}\right\rfloor. \tag{14}$$

Vrijedi da zbrajanje promjene broja operacija do $m$ (uključujući i $m$) iznosi $S(m)$:

$$4_1+5_2+7_3+8_4+\ldots+e_m=4\cdot m+3\cdot\left\lfloor\frac{m^2}{4}\right\rfloor-2\cdot\left\lfloor\frac{m}{2}\right\rfloor. \tag{15}$$

Opći član niza promjene broja operacija $e_m$ računamo zbrajanjem prvog člana $e_1=4$, s nizom promjena promjene do $m$ (uključujući i $m$). Zbog pravilnosti niza, vrijedi formula:

$$e_m=e_1+d_1+2\cdot d_2,$$

gdje $d_1$ označava broj parnih, a $d_2$ broj neparnih brojeva u intervalu $[2,m]$. Izrazimo $d_{1,2}$ preko $m$:

$$d_1=\left\lceil\frac{m-1}{2}\right\rceil,\qquad d_2=\left\lfloor\frac{m-1}{2}\right\rfloor.$$

Uvrštavanjem izraza za $e_m$ u (15) dobivamo jednakost:

$$4_1+5_2+7_3+\ldots+\left(4+\left\lceil\frac{m-1}{2}\right\rceil+2\cdot\left\lfloor\frac{m-1}{2}\right\rfloor\right)_m=4\cdot m+3\cdot\left\lfloor\frac{m^2}{4}\right\rfloor-2\cdot\left\lfloor\frac{m}{2}\right\rfloor. \tag{16}$$

*Dokaz.* Dokazivanjem ove jednakosti indukcijom, dokazujemo ispravnost formule (14).
*Baza indukcije.* Provjerimo da jednakost (16) vrijedi za $m=1$.

Uvrštavanjem $m = 1$ u (16) dobiva se $4 = 4$, što znači da vrijedi za $m = 1$.

*Pretpostavka indukcije.* Pretpostavimo da za neki $s \in \mathbb{N}$ vrijedi:

$$4_1 + 5_2 + 7_3 + \ldots + \left(4 + \left\lceil \frac{s-1}{2} \right\rceil + 2 \cdot \left\lfloor \frac{s-1}{2} \right\rfloor\right)_s = 4 \cdot s + 3 \cdot \left\lfloor \frac{s^2}{4} \right\rfloor - 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor. \tag{17}$$

*Korak indukcije.* Pokažimo da vrijedi i sljedeća jednakost:

$$4_1 + 5_2 + \ldots + \left(4 + \left\lceil \frac{s}{2} \right\rceil + 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor\right)_{s+1} = 4 \cdot (s+1) + 3 \cdot \left\lfloor \frac{(s+1)^2}{4} \right\rfloor - 2 \cdot \left\lfloor \frac{s+1}{2} \right\rfloor.$$

Iz pretpostavke (17) slijedi:

$$4_1 + 5_2 + \ldots + \left(4 + \left\lceil \frac{s}{2} \right\rceil + 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor\right)_{s+1} = 4 \cdot s + 3 \cdot \left\lfloor \frac{s^2}{4} \right\rfloor - 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor + \left(4 + \left\lceil \frac{s}{2} \right\rceil + 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor\right)_{s+1} \tag{18}$$

Na osnovu definicija funkcija pod $\lfloor \cdot \rfloor$ i strop $\lceil \cdot \rceil$ za $n \in \mathbb{Z}$ mogu se dokazati jednakosti:

$$\left\lfloor \frac{n+1}{2} \right\rfloor = \left\lceil \frac{n}{2} \right\rceil,$$

$$\left\lceil \frac{n+1}{2} \right\rceil = \left\lfloor \frac{n}{2} \right\rfloor + 1,$$

$$\left\lfloor \frac{n^2}{4} \right\rfloor = \left\lfloor \frac{n}{2} \right\rfloor \cdot \left\lceil \frac{n}{2} \right\rceil.$$

Primijenimo li ih na izraz (18) dobivamo:

$$\begin{aligned} 4 \cdot s + 3 \cdot \left\lfloor \frac{s^2}{4} \right\rfloor - 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor + \left(4 + \left\lceil \frac{s}{2} \right\rceil + 2 \cdot \left\lfloor \frac{s}{2} \right\rfloor\right)_{s+1} &= 4 \cdot s + 3 \cdot \left\lfloor \frac{s}{2} \right\rfloor \cdot \left\lceil \frac{s}{2} \right\rceil + 4 + \left\lceil \frac{s}{2} \right\rceil \\ &= 4 \cdot (s+1) + \left\lceil \frac{s}{2} \right\rceil \cdot \left(3 \cdot \left\lfloor \frac{s}{2} \right\rfloor + 1\right) \\ &= 4 \cdot (s+1) + \left\lfloor \frac{s+1}{2} \right\rfloor \cdot \left(3 \cdot \left\lceil \frac{s+1}{2} \right\rceil - 2\right) \\ &= 4 \cdot (s+1) + 3 \cdot \left\lfloor \frac{(s+1)^2}{4} \right\rfloor - 2 \cdot \left\lfloor \frac{s+1}{2} \right\rfloor. \end{aligned}$$

Dobivenom jednakosti, po principu potpune indukcije, formula (14) vrijedi za sve $m \in \mathbb{N}$, uz pretpostavku o općenitosti osciliranja vrijednosti niza promjena promjene.

*Program.* Koristeći formulu (14), program 3 može se zapisati u skraćenom obliku, dostupan u poglavlju *Programi* pod br. 4.

## 4.4 DODATAK

Dokazali smo ispravnost izraza (11) te pokazali jednu njegovu izvedenicu, (13), nastalu proširivanjem izraza u sumu. Za kraj ćemo spomenuti još jednu izvedenicu koja proizlazi iz sljedeće jednakosti, pri čemu su $n$ i $k$ prirodni brojevi:

$$\frac{d^k}{dx^k}(x^n) = \frac{n!}{(n-k)!} \cdot x^{n-k}, n \geq k$$

$$\frac{d^k}{dx^k}\left(\frac{x^n}{k!}\right) = \binom{n}{k} \cdot x^{n-k}.$$

Zamjenom odgovarajućih izraza u (13) dolazimo do spomenute izvedenice:

$$x_{n+1} = \frac{\sum_{i=0} \frac{d^{2i}}{dx^{2i}}\left(x_n^{m+1}\right) \cdot \frac{a^i}{(2i)!}}{\sum_{i=0} \frac{d^{2i+1}}{dx^{2i+1}}\left(x_n^{m+1}\right) \cdot \frac{a^i}{(2i+1)!}}.$$

# 5 ZAKLJUČAK

Zaključno, ovim radom smo istaknuli važnost različitih pristupa matematičkim problemima. Konkretno, istraživanjem drugačijeg pristupa problemu računanja kvadratnog korijena došli smo do nekoliko zanimljivih rezultata, kao što su:

(1) Dolazak do opće iteracijske funkcije za približno računanje kvadratnog korijena, te dokaz njene konvergencije.
(2) Izvođenje Babilonske formule.
(3) Konstruiranje formula željenog reda konvergencije...

Ovim rezultatima također potvrđujemo uspješnost pristupa i provedenog istraživanja. Daljnje istraživanje moglo bi uključivati primjenu ovog pristupa na računanje bilo kojeg n-tog korijena, te općenito primjenu pri rješavanju jednadžbi čija su rješenja usko povezana s operacijom korijena.

\clearpage

# 6 PROGRAMI

U ovoj točki prikazani su programski kodovi koji implementiraju postupke korištene u radu. Programi su napisani u programskim jezicima C (programi 1, 3 i 4) i Python (program 2). Njihova je svrha pokazati mogućnost implementacije postupaka na razumljiv i jednostavan način, a ne prikazati optimalan način njihove implementacije.

```c
#include <stdio.h>
#include <math.h>
int main() {
  int i, n; // Brojač iteracija i ukupan broj iteracija
  double a;
  double x, x_new; // Trenutna i sljedeća procjena...

  printf("Unos (a): "); // Unos broja čiji korijen želimo aproksimirat
  scanf("%lf", &a);
  double korijen = sqrt(a); // Referentna vrijednost korijena br. 'a'

  printf("Unos (x_0): ");   // Unos početne procjene
  scanf("%lf", &x);
  printf("Unos broja iteracija: ");
  scanf("%d", &n);

  for (i = 1; i <= n; i++) {
    x_new = 0.5 * (x + (a / x));          // Babilonska formula
    double error = fabs(x_new - korijen); // Računanje greške
    x = x_new;                            // Ažuriranje procjene
    printf("%d. iteracija: %.12lf - %.3e\n", i, x, error);
  }
  return 0; // Kraj programa
}
```

Program 1: *Primjena Babilonske formule (1)*

```python
import numpy as np
import matplotlib.pyplot as plt
import math

# Unos početnih vrijednosti
brF = int(input('Unos broja formula: '))
a = float(input("Unos (a): "))
x_0 = float(input("Unos (x_0): "))
# Unos vrijednosti (k), te računanje aproksimacija
data = [(j, (x := (x_0 if j == 0 else (x * k + a) / (x+k))), abs(math.sqrt(a) - x), k)
        for i in range(brF)
        for k in [float(input(f"Unos (k) za {i+1}. formulu: "))]
        for j in range(6)]

# Razvrstavanje podataka
iteracija, x_stupac, e_stupac, k_stupac = np.array(data).T
k_skup = np.unique(k_stupac)

# Crtanje grafa
plt.figure(figsize=(10, 7))
width = 0.15
position_crtice = np.arange(0, 6)
for i, k in enumerate(k_skup):
  mask = k_stupac == k
  position_trake = iteracija[mask] + i * width
  plt.bar(position_trake, e_stupac[mask], width, label=f'k = {k}')

# Postavljanje oznaka i prikaz grafa
plt.xlabel('Broj iteracije')
plt.ylabel('Greška procjene rješenja')
plt.xticks(position_crtice + width * (len(k_skup) - 1) / 2, position_crtice)
plt.legend()
plt.show()
```

Program 2: *Primjena opće iteracijske funkcije (5)*

```c
#include <stdio.h>
#include <math.h>
// Funkcija za računanje binomnih koeficijenata
void BinomniKoeficijenti(int n, unsigned long long C[]) {
  C[0] = 1;
  for (int k = 0; k < n; k++)
  C[k + 1] = C[k] * (n - k) / (k + 1);}
// Funkcija za konsturiranje i ispis formule te prebrojavanje operacija
void KPI(int binom, int exp_x, int exp_a, int is_last, int *addCount, int *mulCount) {
  printf("%d * x^%d * a^%d", binom, exp_x, exp_a);
  *mulCount += (binom != 1) + (exp_x > 0 ? exp_x : 0) + (exp_a > 0 ? exp_a - 1 : -1);
  if (!is_last) {
    printf(" + "); (*addCount)++; }
}

int main() {
  int addCount = 0, mulCount = 0, divCount = 0; // Brojači operacija
  int m;
  printf("\nUnos (m): ");
  scanf("%d", &m);

  unsigned long long C[m + 1];
  BinomniKoeficijenti(m + 1, C);

  printf("(");
  // Konstruiranje članova brojnika kojih ima p+1
  for (int i = 0, p = ceil(m / 2.0); i <= p; i++)
    KPI(C[i*2], m+1-2*i, i, i==p, &addCount, &mulCount);
  printf(") / ( "); divCount++;
  // Konstruiranje članova nazivnika kojih ima q+1
  for (int i = 0, q = floor(m / 2.0); i <= q; i++)
    KPI(C[i*2+1], m-2*i, i, i==q, &addCount, &mulCount);
  printf(")\n");

  printf("\nBroj operacija zbrajanja: %d\nBroj operacija mnozenja: %d\nBroj operacija dijeljenja: %d\nUkupan broj aritmetickih operacija: %d\n", addCount, mulCount, divCount, addCount + mulCount + divCount);

  return 0;
}
```

Program 3: *Određivanje broja aritmetičkih operacija formule za neki $m$*

```c
#include <stdio.h>
#include <math.h>

int main() {
  int S, n, m;
  printf("Unos broja formula: ");
  scanf("%d", &n);
  for (int i = 1; i <= n; i++) {
    printf("\nUnos (m): ");
    scanf("%d", &m);
    S = 4 * m + 3 * floor(m * m / 4.0) - 2 * floor(m / 2.0); // Formula↩
     ukupnog broja operacija odabrane formule
    printf("\nBroj operacija zbrajanja: %d\nBroj operacija mnozenja: %d↩
    \nBroj operacija dijeljenja: 1\nUkupan broj aritmetickih operacija:↩
     %d\n", m, S - m - 1, S);
  }
  return 0;
}
```

Program 4: *Skraćeni oblik programa 3*

# 7 LITERATURA


[1] Drmač, Z.; Hari, V.; Marušić, M.; Rogina, M.; Singer, S.; Singer, S. (2003.). 'Numerička analiza: Predavanje i vježbe', Sveučilište u Zagrebu, PMF – matematički odsjek, Zagreb, str. 466-471. Preuzeto s: https://web.math.pmf.unizg.hr/ rogina/2001096/num_anal.pdf (Datum pristupa: 2. svibnja 2024.)

[2] 'glatka funkcija | Struna | Hrvatsko strukovno nazivlje'. Preuzeto s: http://struna.ihjj.hr/naziv/glatka-funkcija/32523/#naziv (Datum pristupa: 16. srpnja 2024.)

[3] Kosheleva, O. 'Babylonian Method of Computing the Square Root: Justifications Based on Fuzzy Techniques and on Computational Complexity', University of Texas - Department of Mathematics Education, El Paso. Preuzeto s: https://www.cs.utep.edu/vladik/2009/olg09-05a.pdf (Datum pristupa: 27. svibnja 2024.)

[4] Mladinić, P. (2016). 'RAČUNANJE KORIJENA', Matka, 25(98), str. 116-119. Preuzeto s: https://hrcak.srce.hr/180957 (Datum pristupa: 4. lipnja 2024.)

[5] Zlatić, S. (2013). 'Kaotično ponašanje iteracijskog procesa u Newtonovoj metodi – Newtonov fraktal', Tehnički glasnik, 7(4), str. 347-354. Preuzeto s: https://hrcak.srce.hr/112056 (Datum pristupa: 31. ožujka 2024.)


---

[1]fiksna točka preslikavanja $\varphi(x)$ je točka $x$ za koju vrijedi $\varphi(x) = x$

[2]Red konvergencije označava brzinu kojom se niz približava određenoj vrijednosti. Ako niz konvergira redom $p$, to znači da se greška u svakom koraku smanjuje proporcionalno s $p$-tom potencijom greške u prethodnom koraku.

[3]funkcija je glatka ako ima sve derivacije bilo kojeg reda u svim točkama domene, [2]

ISSN 1334-6083
© 2023 **HMD**