

Numerička matematika

1. predavanje

Saša Singer

PMF – Matematički odsjek, Zagreb

Dobar dan, dobro došli

Sadržaj predavanja

- Uvodna priča o greškama:
 - Problemi numeričke matematike (zašto ona postoji).
 - Pojam greške, apsolutna i relativna greška.
 - Izvori grešaka — model, ulazni podaci (mjerjenje), metoda, zaokruživanje.
 - Ilustracija grešaka na modelnim primjerima.
 - Prikaz brojeva u računalu i greške zaokruživanja (ponavljanje).
 - Greške zaokruživanja osnovnih aritmetičkih operacija.
 - “Širenje” grešaka u aritmetici, stabilne i nestabilne operacije.
 - Opasno ili “katastrofalno” kraćenje.
 - Primjeri iz prakse — posljedice grešaka.

Numerička matematika

Problemi numeričke matematike

U **matematici** postoji niz **problema** koje

● **ne** znamo ili **ne** možemo **egzaktno riješiti**,
tj. **prisiljeni** smo tražiti **približno** rješenje.

Neki klasični “**zadaci**” u numeričkom računanju su:

- rješavanje **sustava linearnih** i **nelinearnih** jednadžbi,
- računanje **integrala**,
- računanje **aproksimacije** neke zadane funkcije (zamjena podataka nekom funkcijom),
- **minimizacija** (maksimizacija) zadane funkcije, uz eventualna **ograničenja** (obično, u domeni),
- rješavanje **diferencijalnih** i **integralnih** jednadžbi ...

Problemi numeričke matematike (nastavak)

Neke probleme čak **znamo** egzaktno riješiti (bar u principu),

☛ poput sustava **linearnih** jednadžbi (ponoviti LA1),
no to **predugo** traje, pa koristimo **računala**.

Međutim, tada imamo **dodatni** problem, jer

☛ računala **ne** računaju **egzaktno**, već **približno!**

Oprez, tada ni **osnovne** aritmetičke operacije **nisu** egzaktne.

Dakle, ključni pojam u **numeric** je

☛ **približna** vrijednost, odnosno, **greška**.

Ciljevi numeričke matematike

U skladu s tim, osnovni **zadatak** numeričke matematike je naći (dati) odgovore na sljedeća pitanja:

- **kako** riješiti neki problem — **metoda**,
- **koliko** je “dobro” izračunato rješenje — **točnost**, **ocjena greške**.

Malo preciznije, za svaku od navedenih klasa problema, treba **proučiti** sljedeće “**teme**” — potprobleme:

1. **Uvjetovanost** problema — **osjetljivost** problema na **greške**, prvenstveno u početnim **podacima** (tzv. teorija perturbacije ili smetnje — vezana uz sam **problem**).
2. **Konstrukcija** standardnih **numeričkih metoda** za **rješavanje** danog problema.

Ciljevi numeričke matematike (nastavak)

Kad jednom “stignemo” do **numeričkih metoda**, treba još **proučiti** sljedeće “**teme**” — potprobleme:

3. **Stabilnost** numeričkih metoda — njihova **osjetljivost** na “smetnje” problema.
4. **Efikasnost** pojedine **numeričke metode** — orijentirano prema implementaciji na **računalu**:
 - broj računskih **operacija** i potreban **memorijski** prostor za rješavanje problema (= **Složenost**).
5. **Točnost** numeričkih metoda, u smislu neke “garancije” točnosti **izračunatog rješenja**.

Ilustracija ovih “potproblema” na primjerima — malo kasnije.

Greške

Greške

Pri **numeričkom** rješavanju nekog problema javljaju se različiti tipovi **grešaka**:

- greške **modela** — svođenje **realnog** problema na neki “**matematički**” problem,
- greške u **ulaznim podacima** (mjerjenja i sl.),
- greške **numeričkih metoda** za rješavanje “**matematičkog**” problema,
- greške “**približnog**” **računanja** — obično su to
 - greške **zaokruživanja** u **aritmetici računala**.

Greške **modela** su “**izvan**” dosega **numeričke matematike**.

- Spadaju u fiziku, kemiju, biologiju, tehniku, ekonomiju, ...

Mjere za grešku

Oznake:

- prava vrijednost — x ,
- izračunata ili približna vrijednost — \hat{x} .

Standardni naziv: \hat{x} je **aproksimacija** za x .

Trenutno, nije bitno **odakle** (iz kojeg skupa) su x i \hat{x} .

- Zamislite da su to “obični” **realni** brojevi — $x, \hat{x} \in \mathbb{R}$.

Mjere za grešku (nastavak)

Apsolutna greška:

- mjeri udaljenost izračunate vrijednosti \hat{x} obzirom na pravu vrijednost x .

Ako imamo vektorski prostor i normu, onda je

- udaljenost = norma razlike.

Dakle, **apsolutna** greška je definirana ovako:

$$E_{\text{abs}}(x, \hat{x}) := |\hat{x} - x|.$$

Često se koristi i oznaka $\Delta x = \hat{x} - x$ (na pr. u analizi), pa je $E_{\text{abs}}(x, \hat{x}) = |\Delta x|$.

Katkad se $\Delta x = \hat{x} - x$ zove “**prava**” greška (predznak bitan).

Mjere za grešku (nastavak)

Primjer. Dojam o “veličini” greške:

- ako smo umjesto 1 izračunali 2, to nam se čini lošije nego
- ako smo umjesto 100 izračunali 101.

Relativna greška:

- mjeri relativnu točnost aproksimacije \hat{x} obzirom na veličinu broja x ,
- na pr. koliko se vodećih znamenki brojeva x i \hat{x} podudara.

Relativna greška definirana je za $x \neq 0$,

$$E_{\text{rel}}(x, \hat{x}) := \frac{|\hat{x} - x|}{|x|}.$$

Često se koristi i oznaka δ_x . Katkad se u nazivniku javlja $|\hat{x}|$.

Mjere za grešku (nastavak)

Ideja relativne greške: ako \hat{x} napišemo kao $\hat{x} = x(1 + \rho)$, onda je njegova **relativna** greška

$$E_{\text{rel}}(x, \hat{x}) := |\rho|.$$

Dakle, **relativna** greška mjeri

- koliko se **faktor** $(1 + \rho)$ apsolutno **razlikuje** od 1.

Sad možemo detaljnije opisati one **četiri** vrste **grešaka**:

- greške **modela**,
- greške u **ulaznim podacima** (mjerenjima),
- greške **metoda za rješavanje modela**,
- greške **aritmetike računala**.

Greške modela

Greške **modela** mogu nastati:

- zbog **zanemarivanja utjecaja nekih sila**,
 - na primjer, zanemarivanje utjecaja **otpora zraka** ili **trenja** (v. primjer),
- zbog **zamjene kompliciranog modela** jednostavnijim,
 - na primjer, sustavi **nelinearnih** običnih ili parcijalnih diferencijalnih jednačbi se **lineariziraju**, da bi se dobilo barem **približno** rješenje,
- zbog upotrebe modela u **graničnim slučajevima**,
 - na primjer, kod **matematičkog** njihala se **$\sin x$** aproksimira s **x** , što vrijedi samo za **male** kutove.

Modelni primjer — Problem gađanja

Primjer. Imamo **top** (ili **haubicu**) u nekoj točki — recimo, **ishodištu**.

- Treba pogoditi **cilj** koji se nalazi u nekoj **drugoj** točki.

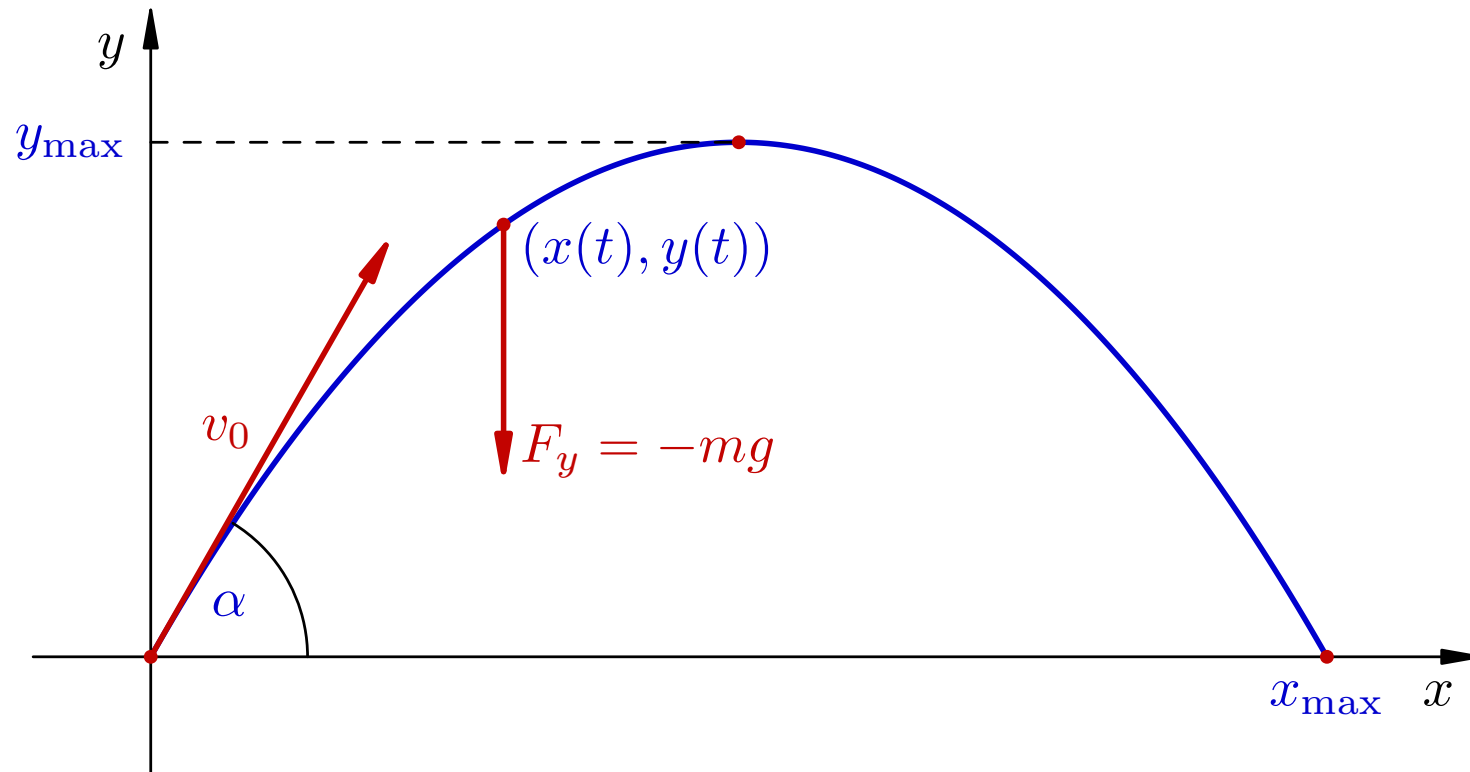
Najjednostavniji model za ovaj problem je poznati **kosi hitac**. Projektil ispaljujemo prema cilju,

- nekom **početnom** brzinom v_0 (vektor),
- pod nekim **kutom** α , obzirom na horizontalnu ravninu.

Slikica (v. sljedeću stranicu)!

Cijela stvar se odvija pod utjecajem **gravitacije** (prema dolje). Ako **zanemarimo otpor** zraka, dobijemo “obični” **kosi hitac**.

Modelni primjer — Slika za kosi hitac



Uzmimo da se cilj nalazi na “istoj visini” — u točki $(x_{\max}, 0)$.

🔴 Udaljenost x_{\max} znamo, a traži se početni kut α .

Modelni primjer — Jednadžba

Osnovna jednadžba je

$$F = ma,$$

gdje je m masa projektila (neće nam trebati na početku), a

- a je **akceleracija** — vektor u **okomitoj** (x, y) -ravnini,
- F je sila **gravitacije**, prema dolje, tj. $F_x = 0$ i $F_y = -mg$.

Gornja jednadžba je **diferencijalna** jednadžba drugog reda u **vremenu**. Ako je $(x(t), y(t))$ **položaj** projektila u danom trenutku, jednadžba ima oblik po komponentama:

$$m \frac{d^2x}{dt^2} = F_x, \quad m \frac{d^2y}{dt^2} = F_y.$$

Akceleracija je **druga** derivacija položaja.

Modelni primjer — Rješenje jednadžbe

Neka je projektil ispaljen u trenutku $t_0 = 0$.

Nakon integracije, za **brzinu** $v =$ **prva** derivacija položaja, imamo jednadžbu

$$mv = F \cdot t + mv_0,$$

ili, po komponentama (masa se skrati)

$$v_x = \frac{dx}{dt} = v_0 \cos \alpha, \quad v_y = \frac{dy}{dt} = v_0 \sin \alpha - gt.$$

Još jednom integriramo (početni položaj je $x_0 = 0$, $y_0 = 0$).

Za **položaj** projektila u trenutku t dobivamo:

$$x(t) = v_0 t \cos \alpha, \quad y(t) = v_0 t \sin \alpha - \frac{1}{2}gt^2.$$

Reklo bi se — znamo sve!

Modelni primjer — Još neke relacije

Jednadžba “putanje” projektila u (x, y) -ravnini je

$$y = x \operatorname{tg} \alpha - \frac{g}{2v_0^2 \cos^2 \alpha} x^2.$$

To je parabola, s otvorom nadolje, koja prolazi kroz ishodište.

Najveća visina projektila je

$$y_{\max} = \frac{(v_0 \sin \alpha)^2}{2g},$$

a maksimalni domet na horizontalnoj x -osi je

$$x_{\max} = \frac{v_0^2 \sin 2\alpha}{g}.$$

Modelni primjer — Stvarnost

Nažalost, s ovim modelom **nećemo** ništa **pogoditi**.

- Fali **otpor** zraka, tlak pada s visinom, vjetrovi i sl.

Praksa:

- Koeficijent **otpora** ovisi o obliku projektila — mjeri se.
- **Izračunate** tablice se **eksperimentalno** “upucavaju” i korigiraju (statistika).
- Primjena u praksi ide **obratno** — znam daljinu, tražim kut.

Na primjer, za obični **kosi hitac**, traženi **kut** je

$$\alpha = \frac{1}{2} \arcsin \frac{x_{\max} g}{v_0^2}.$$

Modelni primjer — Stvarni podaci

Primjer. Za ilustraciju, uzmimo podatke za pravu haubicu kalibra 155 mm, model H155 M65, uz najjače 7. punjenje (maksimalna količina baruta u čahuri).

Početna brzina ispaljene granate je $v_0 = 564 \text{ m/s}$.

Bez otpora zraka, maksimalni domet se postiže za kut $\alpha = 45^\circ = \pi/4$ i iznosi

$$d_{\max} = \frac{564^2 \sin(\pi/2)}{9.81} \approx 32\,425.69 \text{ m.}$$

Stvarni maksimalni domet postiže se za kut $\alpha = 45^\circ 10'$ i iznosi “samo”

$$d_{\max} = 14\,854 \text{ m.}$$

Greške modela (nastavak)

Primjer. Među prvim primjenama jednog od prvih brzih paralelnih računala na svijetu ([ASCI Blue Pacific](#)) bilo je

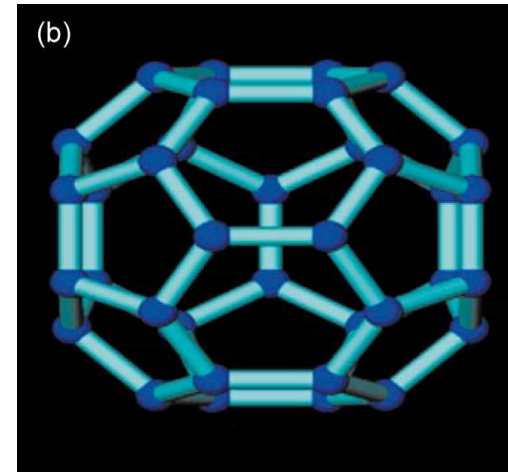
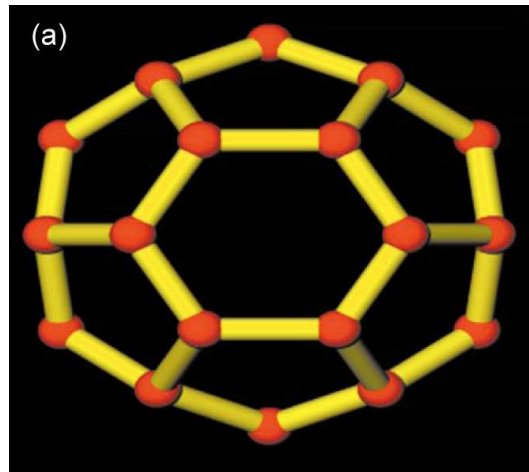
- određivanje trodimenzionalne strukture i elektronskog stanja **ugljik-36 fulerena**.

Primjena spoja je višestruka:

- supravodljivost na visokim temperaturama,
- precizno doziranje lijekova u stanice raka.

Greške modela (nastavak)

Prijašnja istraživanja kvantnih kemičara dala su **dvije** moguće strukture tog spoja, (a) i (b):



Te dvije strukture imaju **različita** kemijska svojstva. Polazno stanje stvari:

- 🔴 **eksperimentalna** mjerenja pokazivala su da je struktura (a) stabilnija,
- 🔴 **teoretičari** su tvrdili da je stabilnija struktura (b).

Greške modela (nastavak)

Prijašnja računanja,

- zbog pojednostavljivanja i interpolacije, kao odgovor, davala su prednost “teoretskoj” strukturi (b).

Definitivan odgovor,

- proveden računanjem bez pojednostavljivanja, pokazao je da je struktura (a) stabilnija.

Poanta: pretjerano pojednostavljenje bilo čega (modela, metode ili algoritma) može dovesti

- do pogrešnih rezultata!

Dobivene rezultate je zdravo provjeriti — eksperimentom ili točnijim računom!

Greške u ulaznim podacima

Greške u **ulaznim podacima** javljaju se zbog

- **nemogućnosti** ili **besmislenosti** točnog mjerenja (Heisenbergove relacije neodređenosti).
- Na primjer, tjelesna temperatura se obično mjeri na **desetinku** stupnja Celziosa točno. Pacijent je podjednako **loše** ako ima temperaturu **39.5°C** ili **39.513462°C**.

Bitno **praktično** pitanje:

- Mogu li **male** greške u ulaznim podacima bitno **povećati** grešku rezultata?

Nažalost **MOGU!**

- Takvi problemi zovu se **loše uvjetovani problemi**.

Greške u ulaznim podacima (nastavak)

Primjer. Zadana su dva sustava linearnih jednadžbi — recimo, umjesto ispravnih (**prvih**) koeficijanata, **izmjerili** smo **druge**:

$$2x + 6y = 8$$

$$2x + 6.0001y = 8.0001,$$

i

$$2x + 6y = 8$$

$$2x + 5.99999y = 8.00002.$$

Samo **druga** jednadžba se “malo” **promijenila**.

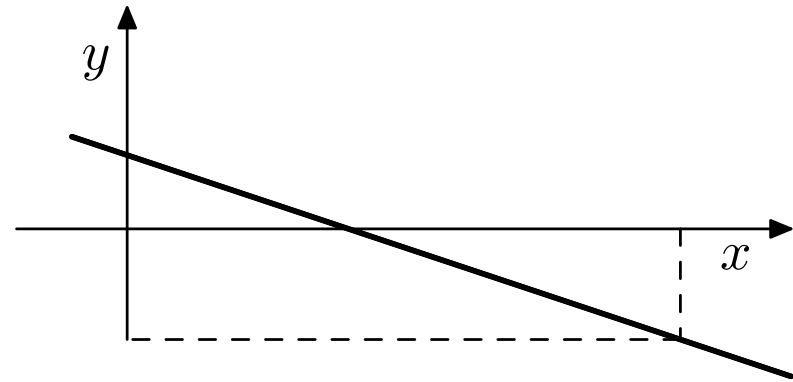
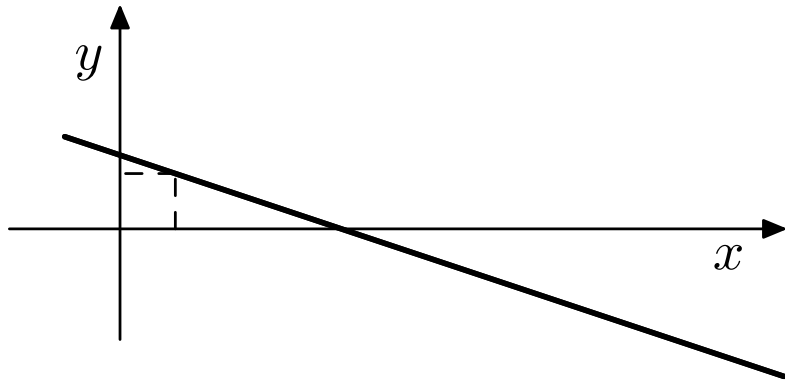
Perturbacije **koeficijenata** su reda veličine 10^{-4} .

Je li se **rezultat**, također, promijenio za red veličine 10^{-4} ?

Greške u ulaznim podacima (nastavak)

- Rješenje prvog problema: $x = 1, y = 1$.
- Rješenje drugog problema: $x = 10, y = -2$.

Grafovi presjecišta dva pravca za prvi i drugi sustav:



Koja dva pravca???

- U oba sustava, pravci su “skoro” paralelni, pa se njihovi grafovi ne razlikuju na slikama — i baš tu je problem!

Greške metoda za rješavanje problema

Najčešće nastaju kad se nešto **beskonačno** zamjenjuje nečim **konačnim**. Razlikujemo **dvije** kategorije:

- **greške diskretizacije**, koje nastaju
 - zamjenom **kontinuum**a (neprebrojiv skup) **konačnim diskretnim** skupom točaka,
 - ili “**beskonačno**” malu veličinu h ili $\varepsilon \rightarrow 0$ zamijenjujemo nekim “**konačno**” malim brojem;
- **greške odbacivanja**, koje nastaju
 - “**rezanjem**” **beskonačnog** niza ili reda na **konačni** niz ili sumu, tj. odbacujemo ostatak niza ili reda.

Ovo je zamjena **beskonačnog diskretnog** skupa (prebrojiv skup, poput \mathbb{N}) **konačnim** skupom.

Greške metoda za rješavanje problema (nast.)

Tipični primjeri greške diskretizacije:

- aproksimacija funkcije f na $[a, b]$, vrijednostima te funkcije na konačnom skupu točaka (tzv. mreži) $\{x_1, \dots, x_n\} \subset [a, b]$,
- aproksimacija derivacije funkcije f u nekoj točki x . Po definiciji je

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h},$$

a za približnu vrijednost uzmemo dovoljno mali $h \neq 0$ i

$$f'(x) \approx \frac{\Delta f}{\Delta x} = \frac{f(x+h) - f(x)}{h}.$$

Greške metoda za rješavanje problema (nast.)

Tipični primjeri greške odbacivanja:

- zaustavljanje iterativnih procesa nakon dovoljno velikog broja n iteracija (recimo, kod računanja nultočaka funkcije);
- zamjena beskonačne sume konačnom — kad greška postane dovoljno mala (recimo, kod sumiranja Taylorovih redova — v. sljedeći primjer).

Taylorov red, Taylorov polinom, ...

Za dovoljno glatku funkciju f , Taylorov red oko točke x_0

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k$$

možemo **aproksimirati** Taylorovim polinomom p

$$f(x) = p(x) + R_{n+1}(x), \quad p(x) = \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k,$$

pri čemu je

$$R_{n+1}(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x - x_0)^{n+1}.$$

greška odbacivanja, a ξ neki broj između x_0 i x . Grešku $R_{n+1}(x)$ obično ocjenjujemo po apsolutnoj vrijednosti.

Taylorov red, Taylorov polinom, ... (nastavak)

Primjer.

- Funkcije e^x i $\sin x$ imaju Taylorove redove oko točke $x_0 = 0$, koji **konvergiraju** za proizvoljan $x \in \mathbb{R}$.
- Zbrajanjem dovoljno mnogo članova tih redova, možemo, barem u principu, po volji dobro **aproksimirati** vrijednosti funkcija e^x i $\sin x$.
- Traženi Taylorovi polinomi s **istim brojem** članova (ali **ne** istog stupnja) su

$$e^x \approx \sum_{k=0}^n \frac{x^k}{k!}, \quad \sin x \approx \sum_{k=0}^n \frac{(-1)^k x^{2k+1}}{(2k+1)!}.$$

Taylorov red, Taylorov polinom, ... (nastavak)

Za grešku odbacivanja trebaju nam derivacije:

$$(e^x)^{(n)} = e^x, \quad (\sin x)^{(n)} = \sin \left(x + \frac{n\pi}{2} \right),$$

pa su pripadne greške odbacivanja

$$R_{n+1}(x) = \frac{e^\xi x^{n+1}}{(n+1)!}, \quad R_{2n+3}(x) = \frac{\sin \left(\xi + \frac{2n+3}{2} \pi \right) x^{2n+3}}{(2n+3)!},$$

Pretpostavimo sada da je $x > 0$. Iz $\xi \leq x$ slijedi $e^\xi \leq e^x$, pa dobivamo

$$|R_{n+1}(x)| \leq \frac{e^x x^{n+1}}{(n+1)!}, \quad |R_{2n+3}(x)| \leq \frac{x^{2n+3}}{(2n+3)!}.$$

Taylorov red, Taylorov polinom, ... (nastavak)

Zbrojimo li članove reda sve dok apsolutna vrijednost **prvog odbačenog** člana ne padne ispod **zadane točnosti** $\varepsilon > 0$, napravili smo **grešku odbacivanja** manju ili jednaku

$$\begin{cases} e^x \varepsilon, & \text{za } e^x, \\ \varepsilon, & \text{za } \sin x. \end{cases}$$

U **prvom** slučaju očekujemo

● **malu relativnu** grešku,

a u **drugom** slučaju očekujemo

● **malu apsolutnu** grešku.

Provjerimo to eksperimentalno — u **aritmetici računala!**

Cijelo računanje provedeno je u standardnom tipu **double**.

Kako se računaju članovi reda?

Članove reda računamo **rekurzivno** — novi iz prethodnog.

Za \exp :

$$\text{član}_0 = \frac{x^0}{0!} = 1,$$

$$\text{član}_k = \frac{x^k}{k!} = \frac{x \cdot x^{k-1}}{k \cdot (k-1)!} = \frac{x}{k} \cdot \text{član}_{k-1}, \quad k \geq 1.$$

Za \sin :

$$\text{član}_0 = \frac{(-1)^0 x^1}{1!} = x,$$

$$\begin{aligned} \text{član}_k &= \frac{(-1)^k x^{2k+1}}{(2k+1)!} = \frac{-x^2 \cdot (-1)^{k-1} x^{2k-1}}{(2k+1) \cdot 2k \cdot (2k-1)!} \\ &= \frac{-x^2}{2k(2k+1)} \cdot \text{član}_{k-1}, \quad k \geq 1. \end{aligned}$$

Red za eksponencijalnu funkciju, $x = 12\pi$

Za $x = 12\pi$ i $\varepsilon = 5 \cdot 10^{-16}$ trebamo 132 člana reda ($n = 131$)

$$|\text{greška odbacivanja}| \leq 2.5101 \cdot 10^0$$

$$|\text{maksimalni član}| = 1.5329 \cdot 10^{15} \quad (n = 37).$$

Dobivamo:

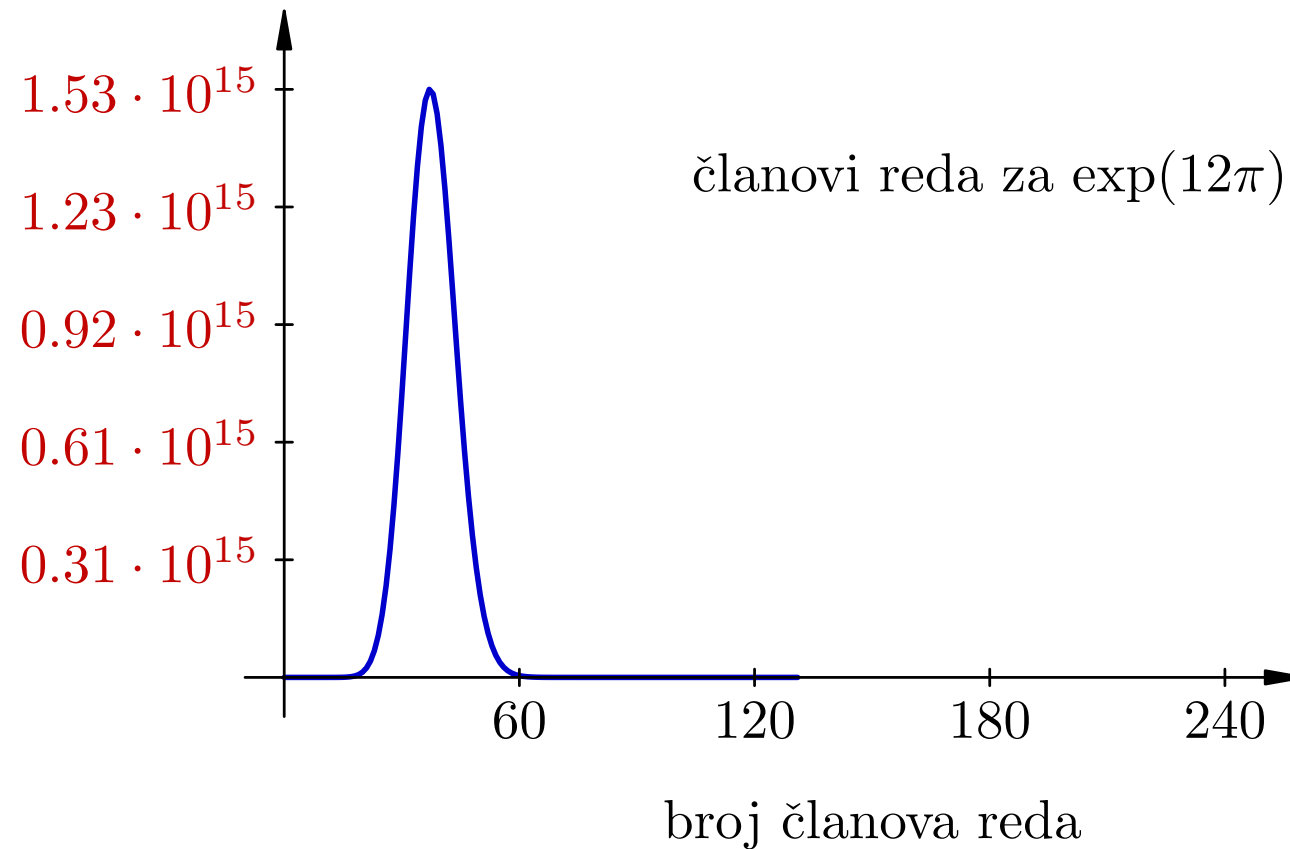
$$\exp(12\pi)_{\text{funkcija}} = 2.3578503968558192 \cdot 10^{16}$$

$$\exp(12\pi)_{\text{Taylor}} = 2.3578503968558196 \cdot 10^{16}$$

$$\text{prava greška} = -4.0000000000000000 \cdot 10^0$$

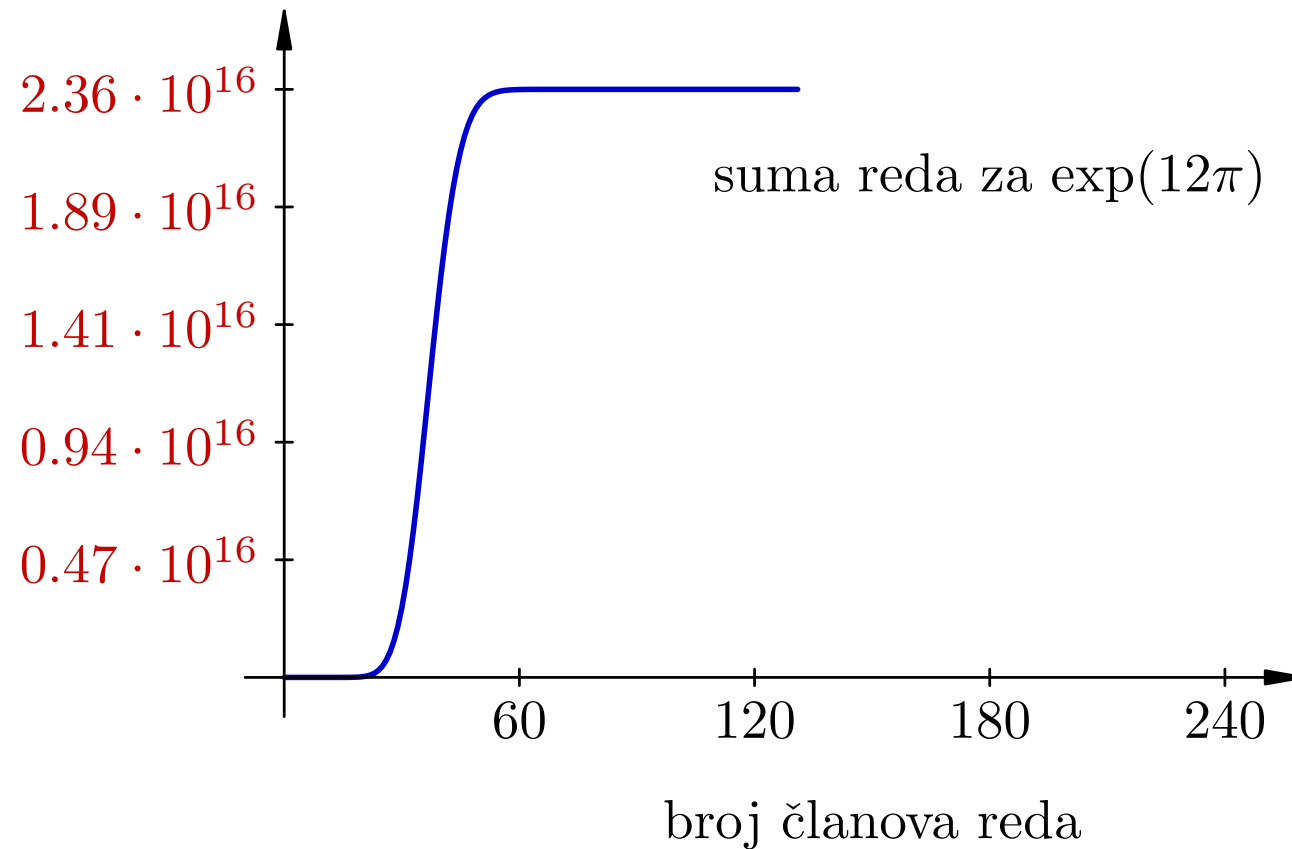
$$\text{relativna greška} = 1.6964604732064335 \cdot 10^{-16}.$$

Članovi reda za $\exp(12\pi)$



Svi članovi imaju **isti** predznak.

Suma reda za $\exp(12\pi)$



Suma stalno **raste**, dok se ne **stabilizira**.

Red za eksponencijalnu funkciju, $x = 24\pi$

Za $x = 24\pi$ i $\varepsilon = 5 \cdot 10^{-16}$ trebamo 236 članova reda ($n = 235$)

$$|\text{greška odbacivanja}| \leq 5.0445 \cdot 10^{16}$$

$$|\text{maksimalni član}| = 2.5555 \cdot 10^{31} \quad (n = 75).$$

Dobivamo:

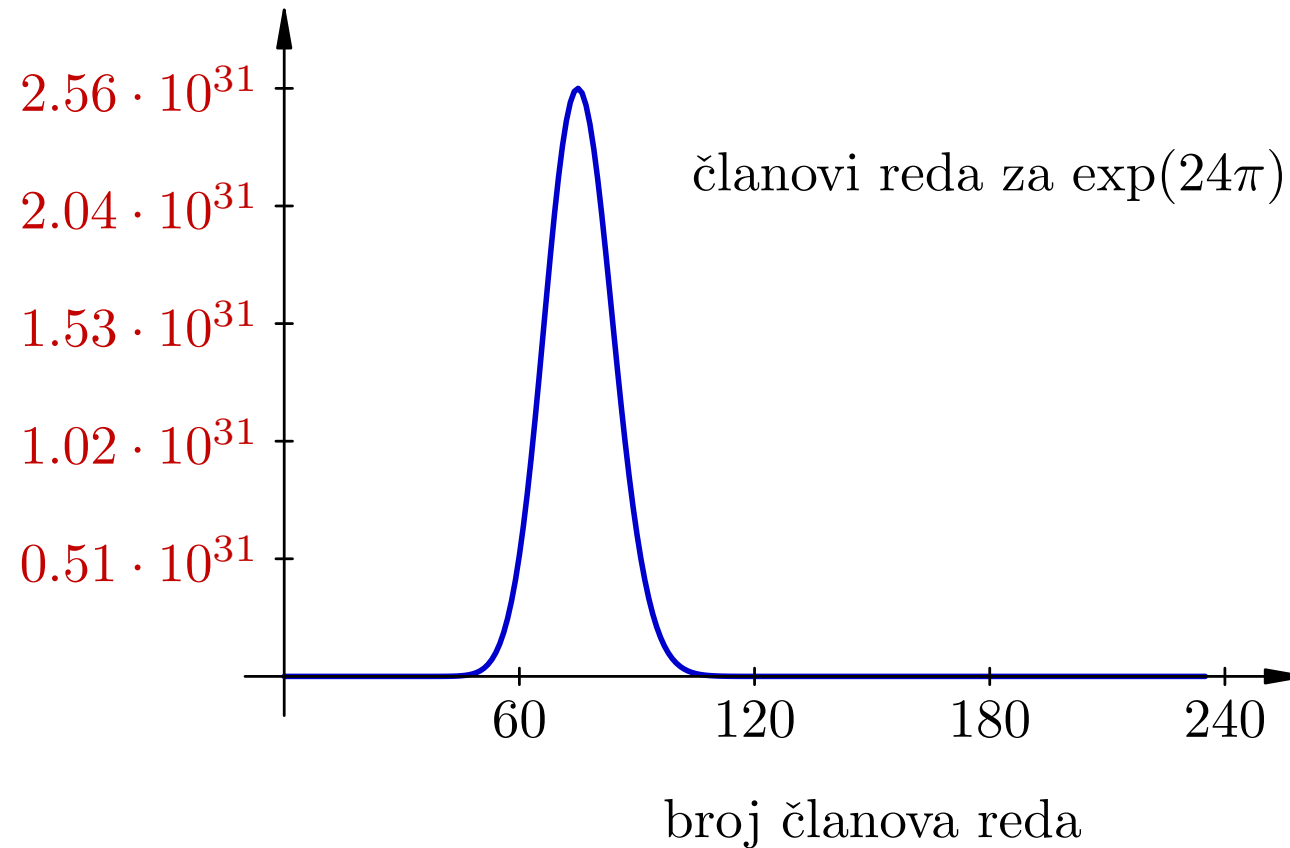
$$\exp(24\pi)_{\text{funkcija}} = 5.5594584939531437 \cdot 10^{32}$$

$$\exp(24\pi)_{\text{Taylor}} = 5.5594584939531445 \cdot 10^{32}$$

$$\text{prava greška} = -7.2057594037927936 \cdot 10^{16}$$

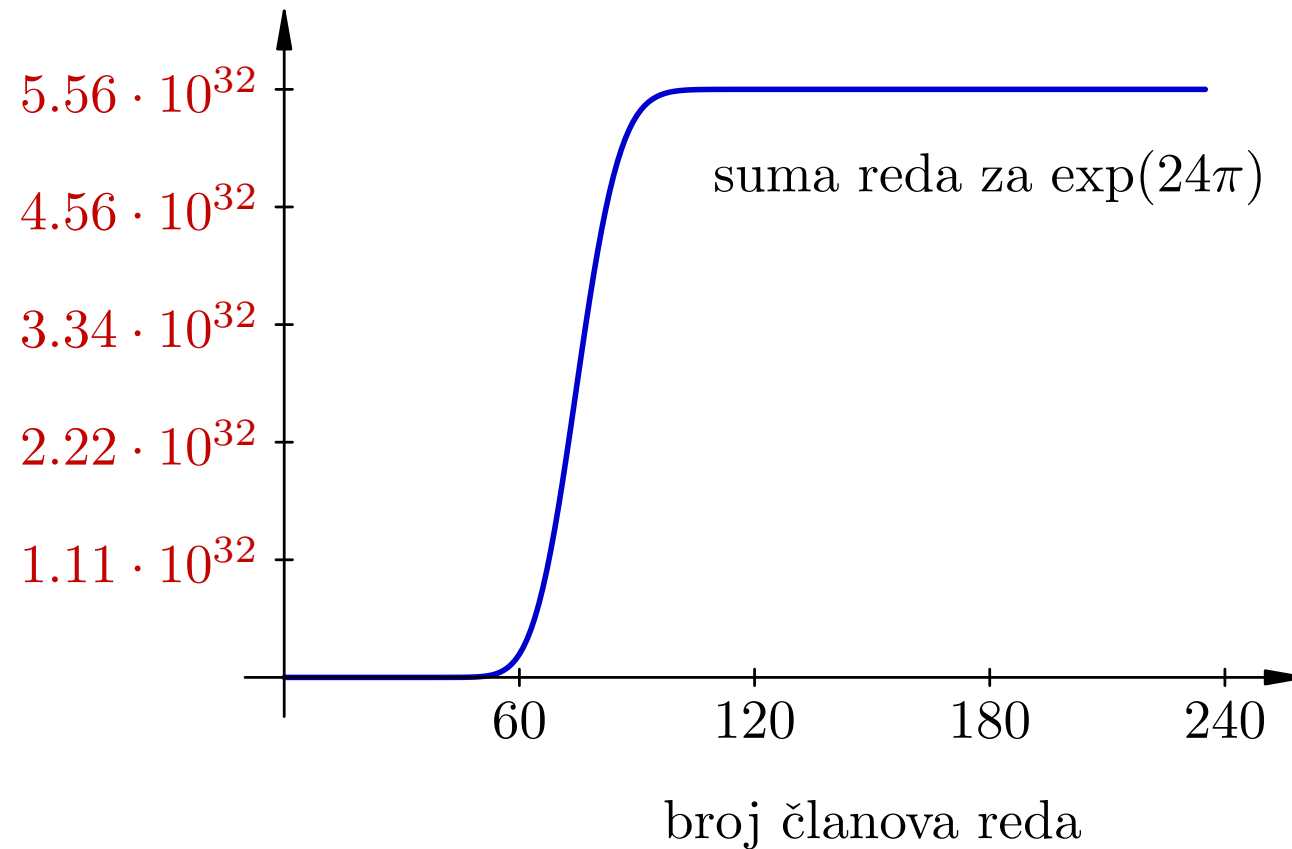
$$\text{relativna greška} = 1.2961261266057264 \cdot 10^{-16}.$$

Članovi reda za $\exp(24\pi)$



Svi članovi imaju **isti** predznak.

Suma reda za $\exp(24\pi)$



Suma stalno **raste**, dok se ne **stabilizira**.

Red za funkciju sinus, $x = 12\pi$

Za $x = 12\pi$ i $\varepsilon = 5 \cdot 10^{-16}$ trebamo 66 članova reda ($n = 131$)

$$|\text{greška odbacivanja}| \leq 3.0175 \cdot 10^{-17}$$

$$|\text{maksimalni član}| = 1.5329 \cdot 10^{15} \quad (n = 37).$$

Dobivamo:

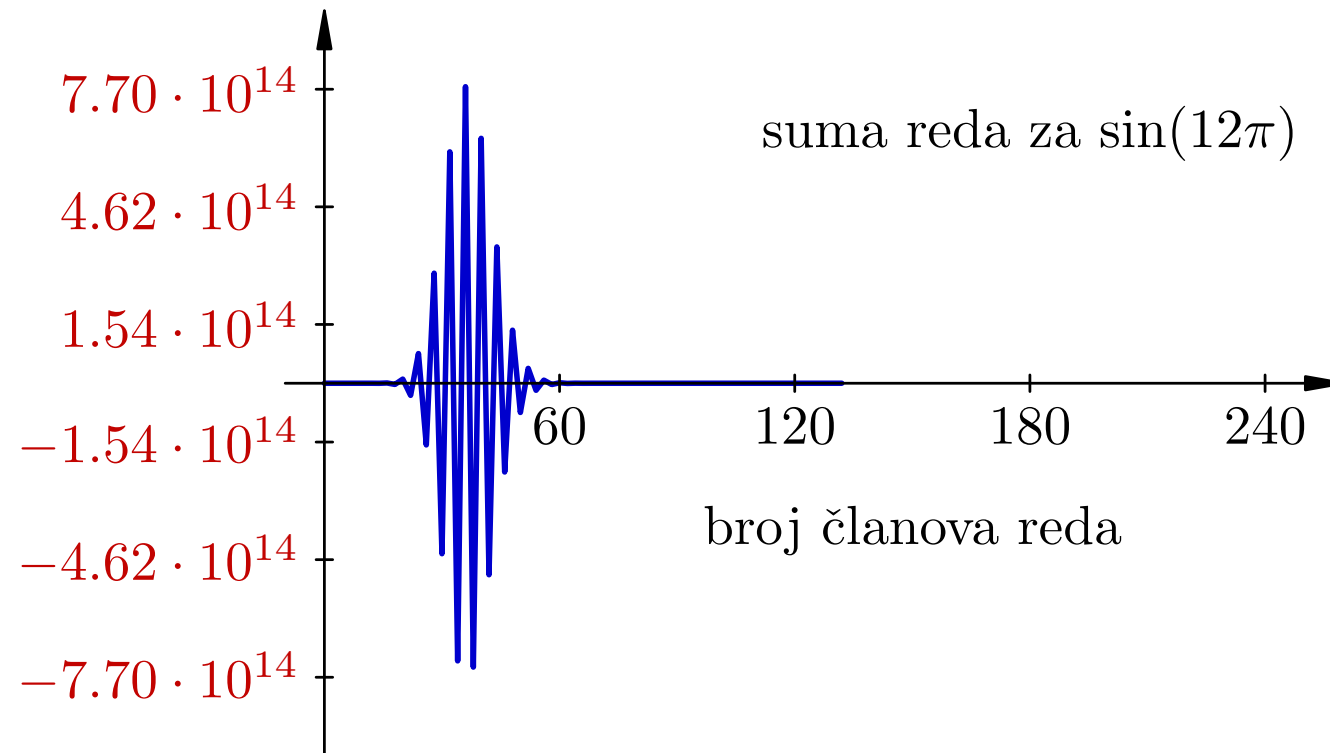
$$\sin(12\pi)_{\text{funkcija}} = -1.4695276245868527 \cdot 10^{-15}$$

$$\sin(12\pi)_{\text{Taylor}} = -4.1381632107344454 \cdot 10^{-2}$$

$$\text{prava greška} = 4.1381632107342983 \cdot 10^{-2}$$

$$\text{relativna greška} = 2.8159819124854586 \cdot 10^{13}.$$

Suma reda za $\sin(12\pi)$



Suma naglo **naraste**, a zatim se **skrati**, dok se ne **stabilizira**.

Posljedica: **gubitak točnosti** — relativno obzirom na **najveći međurezultat**.

Red za funkciju sinus, $x = 24\pi$

Za $x = 24\pi$ i $\varepsilon = 5 \cdot 10^{-16}$ trebamo 118 članova reda ($n = 235$)

$$|\text{greška odbacivanja}| \leq 2.8867 \cdot 10^{-17}$$

$$|\text{maksimalni član}| = 2.5555 \cdot 10^{31} \quad (n = 75).$$

Dobivamo:

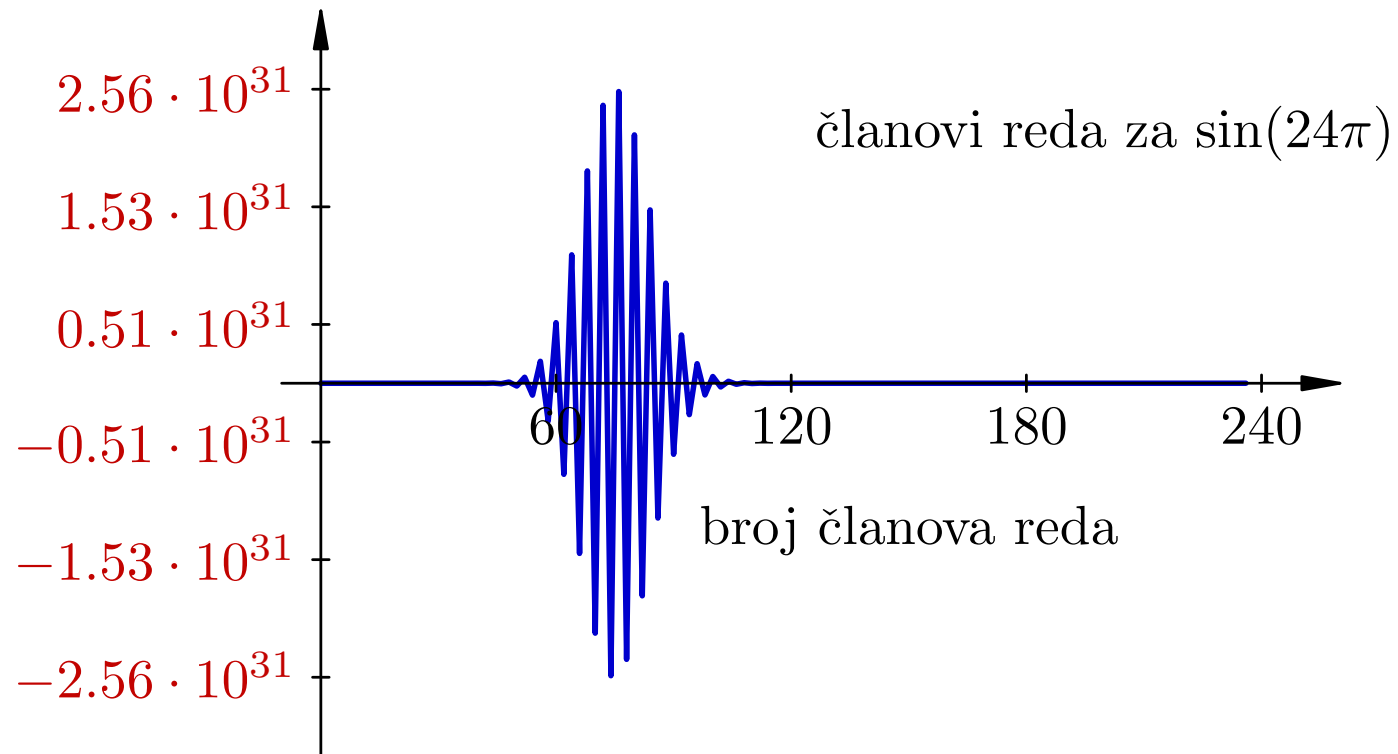
$$\sin(24\pi)_{\text{funkcija}} = -2.9390552491737054 \cdot 10^{-15}$$

$$\sin(24\pi)_{\text{Taylor}} = 3.6199983145905898 \cdot 10^{13}$$

$$\text{prava greška} = -3.6199983145905898 \cdot 10^{13}$$

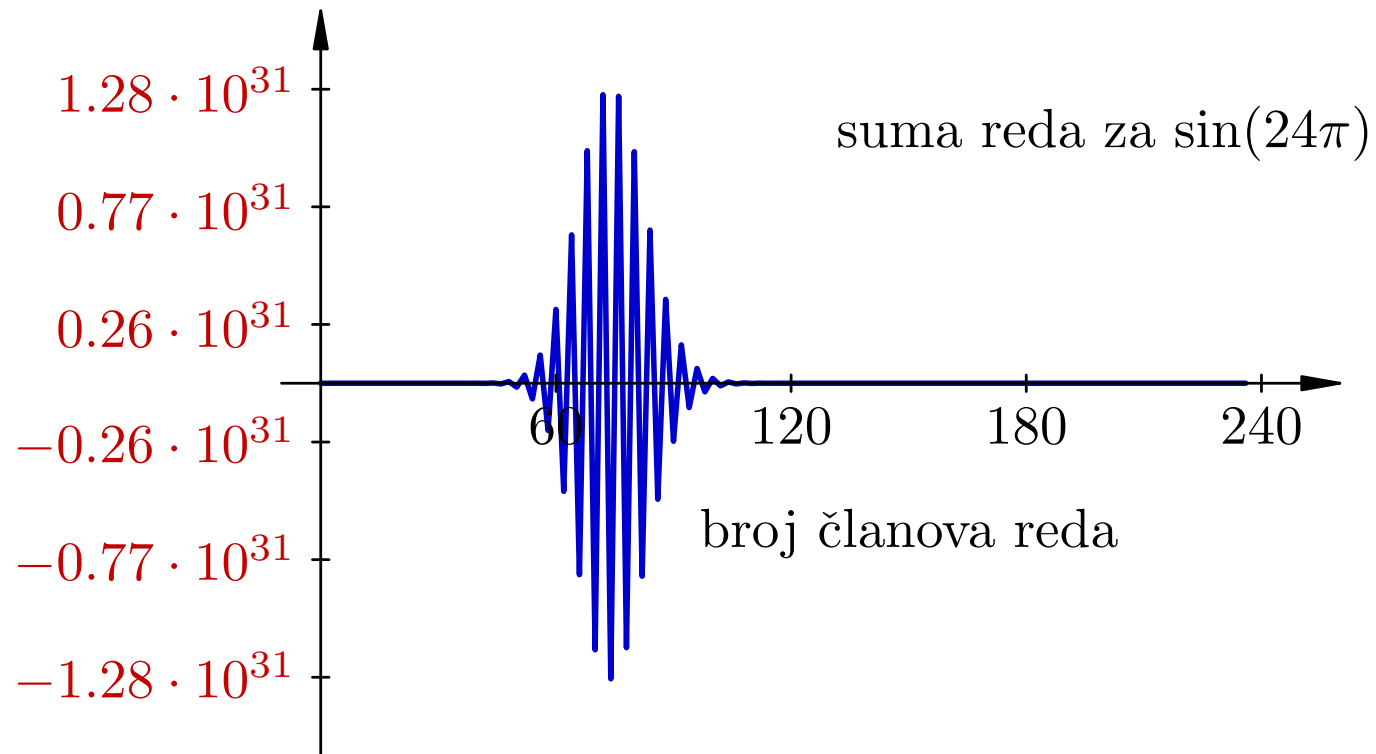
$$\text{relativna greška} = 1.2316877389794990 \cdot 10^{28}.$$

Članovi reda za $\sin(24\pi)$



Članovi **alterniraju** po predznaku.

Suma reda za $\sin(24\pi)$



Suma naglo **naraste**, a zatim se **skrati**, dok se ne **stabilizira**.

Posljedica: **gubitak točnosti** — relativno obzirom na **najveći međurezultat**.

Prikaz brojeva u računalu i greške zaokruživanja

Tipovi brojeva u računalu

U računalu postoje dva bitno različita tipa brojeva:

- cijeli brojevi
- realni brojevi.

Oba skupa su **konačni podskupovi** odgovarajućih skupova \mathbb{Z} i \mathbb{R} u matematici.

Kao **baza** za prikaz **oba** tipa koristi se baza **2**.

Podtipovi — ovisno o broju bitova n predviđenih za prikaz odgovarajuće vrste brojeva.

Cijeli brojevi — sažetak

Cijeli brojevi bez predznaka:

$$\mathbb{Z}_{2^n} = \{ 0, 1, 2, \dots, 2^n - 2, 2^n - 1 \}.$$

Cijeli brojevi s predznakom:

$$\mathbb{Z}_{2^n}^- = \{ -2^{n-1}, -2^{n-1} + 1, \dots, -2, -1, \\ 0, 1, \dots, 2^{n-1} - 2, 2^{n-1} - 1 \}.$$

Aritmetika cijelih brojeva je modularna aritmetika modulo 2^n :

- operacije $+$, $-$ i \cdot daju cjelobrojni rezultat modulo 2^n ,
- operacije cjelobrojnog dijeljenja s ostatkom daju zaokruženi racionalni kvocijent (prema nuli) i pripadni ostatak (s predznakom prvog argumenta).

Oprez: $n!$ u cjelobrojnoj aritmetici — $50! = 0$ (modulo 2^{32}).

Prikaz realnih brojeva

sažetak

Realni brojevi — prikaz

Skup svih realnih brojeva prikazivih u računalu je konačan, a parametriziramo ga duljinom mantise (t) i eksponenta (w) i označavamo s $\mathbb{R}(t, w)$.

mantisa (signifikand)

1.	b_{-1}	b_{-2}	\cdots	b_{-t}
----	----------	----------	----------	----------

eksponent (karakteristika)

e_{w-1}	e_{w-2}	\cdots	e_1	e_0
-----------	-----------	----------	-------	-------

Oznaka: preciznost $p := t + 1$ — to je ukupni broj vodećih značajnih bitova cijele mantise (zajedno s vodećim 1).

Ne može se svaki realni broj egzaktano spremiti u računalo.

Neka je broj $x \in \mathbb{R}$ unutar prikazivog raspona i

$$x = \pm \left(1 + \sum_{i=1}^{\infty} b_{-i} \cdot 2^{-i} \right) \cdot 2^e.$$

Realni brojevi — zaokruživanje

Ako **razlomljeni** dio mantise ima **više** od t znamenki, bit će spremljena **aproksimacija** tog broja $fl(x) \in \mathbb{R}(t, w)$, koja se može prikazati kao

$$fl(x) = \pm \left(1 + \sum_{i=1}^t b_{-i}^* \cdot 2^{-i} \right) \cdot 2^{e^*}.$$

Slično kao kod decimalne aritmetike,

- ako je **prva** odbačena znamenka **1**, broj zaokružujemo **nagore**,
- a ako je **0**, **nadolje**.

Time smo napravili **apsolutnu grešku** manju ili jednaku od “**pola zadnjeg prikazivog bita**”, tj. $2^{-t-1+e} = 2^{-p+e}$.

Relativna greška zaokruživanja

Gledajući **relativno**, greška je manja ili jednaka

$$\left| \frac{x - fl(x)}{x} \right| \leq \frac{2^{-t-1+e}}{2^0 \cdot 2^e} = 2^{-t-1} = 2^{-p},$$

tj. imamo vrlo **malu** relativnu grešku.

Veličinu $2^{-t-1} = 2^{-p}$ zovemo **jedinična greška zaokruživanja** (engl. unit roundoff) i uobičajeno označavamo s u .

Za $x \in \mathbb{R}$ unutar **prikazivog** raspona, umjesto x sprema se **zaokruženi** broj $fl(x) \in \mathbb{R}(t, w)$ i vrijedi

$$fl(x) = (1 + \varepsilon)x, \quad |\varepsilon| \leq u,$$

gdje je ε **relativna** greška napravljena tim zaokruživanjem.

Standardni tipovi realnih brojeva — IEEE 754

Novi standard IEEE 754-2008 standard ima sljedeće tipove za prikaz realnih brojeva:

ime tipa	binary32	binary64	binary128
duljina u bitovima	32	64	128
$t =$	23	52	112
$w =$	8	11	15
$u = 2^{-p}$	2^{-24}	2^{-53}	2^{-113}
$u \approx$	$5.96 \cdot 10^{-8}$	$1.11 \cdot 10^{-16}$	$9.63 \cdot 10^{-35}$
raspon brojeva \approx	$10^{\pm 38}$	$10^{\pm 308}$	$10^{\pm 4932}$

Najveći tip `binary128` još uvijek **ne postoji** u većini procesora.

Standardni tipovi realnih brojeva — extended

Većina **PC** procesora još uvijek ima posebni dio — tzv. **FPU** (engl. Floating-Point Unit). On **stvarno** koristi

- tip **extended** iz **starog** standarda, koji odgovara tipu **extended binary64** u **novom IEEE 754-2008** standardu.

Dio primjera koje ćete vidjeti napravljen je baš u **tom tipu!**

ime tipa	extended
duljina u bitovima	80
$t + 1 =$	$63 + 1$
$w =$	15
$u = 2^{-p}$	2^{-64}
$u \approx$	$5.42 \cdot 10^{-20}$
raspon brojeva \approx	$10^{\pm 4932}$

Realna aritmetika računala (IEEE standard)

Realna aritmetika računala — standard

Realna aritmetika računala nije egzaktna!

Razlog:

- Rezultat svake operacije mora biti prikaziv,
- pa dolazi do zaokruživanja.

Standard IEEE 754-2008 za realnu aritmetiku računala propisuje da za sve četiri osnovne aritmetičke operacije vrijedi

- ista ocjena greške zaokruživanja kao i za prikaz brojeva,
- tj. da izračunati rezultat ima malu relativnu grešku.

Isto vrijedi i za neke matematičke funkcije, poput $\sqrt{\quad}$, ali ne mora vrijediti za sve funkcije (na pr. za \sin oko 0, ili \ln oko 1).

Standard iz 2008. g. to preporučuje, ali (zasad) ne zahtijeva.

Realna aritmetika računala — zaokruživanje

Neka je \circ bilo koja od aritmetičkih operacija $+$, $-$, $*$, $/$, i neka su x i y prikazivi operandi (drugih, ionako, nema u računalu).

- Ako su x i y u dozvoljenom, tj. normaliziranom rasponu,
- i ako se egzaktni rezultat $x \circ y$, također, nalazi u normaliziranom rasponu (ne mora biti prikaziv),

za računalom izračunati (prikazivi) rezultat $fl(x \circ y)$ onda vrijedi

$$fl(x \circ y) = (1 + \varepsilon)(x \circ y), \quad |\varepsilon| \leq u,$$

gdje je u jedinična greška zaokruživanja za dani tip brojeva. Ova ocjena odgovara zaokruživanju egaktnog rezultata!

Prava relativna greška ε ovisi o: x , y , operaciji \circ , i stvarnoj realizaciji aritmetike računala.

Posljedice zaokruživanja u realnoj aritmetici

Napomena. Bez pretpostavki o **normaliziranom** rasponu, prethodni rezultat **ne vrijedi** — greška može biti **puno veća!**

Zbog **zaokruživanja**, u realnoj aritmetici računala, nažalost,

- **ne vrijede** uobičajeni **zakoni** za aritmetičke operacije na skupu \mathbb{R} .

Na primjer, za aritmetičke operacije u **računalu**

- **nema asocijativnosti** zbrajanja i množenja,
- **nema distributivnosti** množenja prema zbrajanju.

Dakle, **poredak izvršavanja operacija** je **bitan!**

Zapravo, **jedino** standardno pravilo koje **vrijedi** je

- **komutativnost** za zbrajanje i za množenje.

Širenje grešaka zaokruživanja

Širenje grešaka zaokruživanja

Vidimo da gotovo **svaki** izračunati rezultat ima neku **grešku**.
Osim toga,

- zaokruživanje se vrši nakon **svake pojedine operacije**.

Najlakše je stvar zamišljati kao da zaokruživanje ide “na kraju” operacije, iako je ono “dio operacije”.

Kad imamo **puno** aritmetičkih operacija (inače nam računalo ne treba), dolazi do tzv.

- akumulacije** ili **širenja** grešaka zaokruživanja.

Malo pogrešni rezultati (možda već od čitanja), ulaze u operacije, koje opet malo griješe, i tako redom ...

- greške** se “**šire**” kroz **sve što računamo!**

Opasne i bezopasne operacije — sažetak

Jedina opasna operacija — kad rezultat može imati veliku relativnu grešku, je

- oduzimanje bliskih brojeva,
- i to samo kad polazni operandi već imaju neku grešku (samo oduzimanje je tada, najčešće, egzaktno).

Ovaj fenomen zove se opasno ili katastrofalno kraćenje.

Sve ostale operacije su bezopasne — relativna greška rezultata ne raste pretjerano. Posebno,

- dijeljenje malim brojem nije opasno,
- osim kad je mali broj nastao (ranijim) kraćenjem.

Nažalost, u nekim knjigama piše suprotno — i pogrešno.

Širenje grešaka u aritmetici

Širenje grešaka u aritmetici

Za analizu širenja grešaka u aritmetici, treba pogledati

- što se događa s greškama u rezultatu,
- kad imamo greške u operandima.

Prvo u egzaktnoj aritmetici, a onda i u aritmetici računala.

Pretpostavimo onda da su polazni podaci (ili operandi) x i y malo perturbirani, s pripadnim relativnim greškama ε_x i ε_y .

Koje su operacije opasne (ako takvih ima), ako nam je aritmetika egzaktna, a operandi su $x(1 + \varepsilon_x)$ i $y(1 + \varepsilon_y)$?

Treba ocijeniti relativnu grešku ε_o rezultata operacije \circ

$$(x \circ y) (1 + \varepsilon_o) := [x (1 + \varepsilon_x)] \circ [y (1 + \varepsilon_y)].$$

Širenje grešaka u aritmetici (nastavak)

Naravno, za početak, moramo nešto **pretpostaviti** o ε_x i ε_y .

Što smatramo **malom** relativnom perturbacijom?

- Svakako **mora** biti $|\varepsilon_x|, |\varepsilon_y| < 1$, inače perturbacijom **gubimo predznak** operanda.

Međutim, to nije dovoljno za neki razuman rezultat.

- Stvarno **očekujemo** $|\varepsilon_x|, |\varepsilon_y| \leq c \ll 1$, tako da imamo barem **nekoliko točnih znamenki** u perturbiranim operandima. Na pr., $c = 10^{-1}$ (jedna točna znamenka).
- **Idealno**, u računalu je $|\varepsilon_x|, |\varepsilon_y| \leq u$, tj. kao da smo oba operanda **samo spremili u memoriju** računala (jedna greška zaokruživanja).

Širenje grešaka kod množenja

Množenje je bezopasno (benigno), jer vrijedi

$$\begin{aligned}(x * y) (1 + \varepsilon_*) &:= [x (1 + \varepsilon_x)] * [y (1 + \varepsilon_y)] \\ &= xy (1 + \varepsilon_x + \varepsilon_y + \varepsilon_x \varepsilon_y),\end{aligned}$$

kad stvar napišemo bez nepotrebnih zagrada i *. Onda je

$$\varepsilon_* = \varepsilon_x + \varepsilon_y + \varepsilon_x \varepsilon_y \approx \varepsilon_x + \varepsilon_y,$$

ako su $|\varepsilon_x|$ i $|\varepsilon_y|$ dovoljno mali da $\varepsilon_x \varepsilon_y$ možemo zanemariti.

Dakle, relativna greška se samo zbraja.

U idealnom slučaju $|\varepsilon_x|, |\varepsilon_y| \leq u$, dobivamo približnu ocjenu relativne greške $|\varepsilon_*| \leq 2u$ (do na u^2), ili, na pr., $|\varepsilon_*| \leq 2.01u$.

Širenje grešaka kod dijeljenja

Dijeljenje je, također, bezopasno (benigno), samo je zaključak malo dulji. Na početku je

$$(x / y) (1 + \varepsilon_{/}) := [x (1 + \varepsilon_x)] / [y (1 + \varepsilon_y)] = \frac{x (1 + \varepsilon_x)}{y (1 + \varepsilon_y)}.$$

Ako su $|\varepsilon_x|$ i $|\varepsilon_y|$ dovoljno mali da sve možemo linearizirati (tj. zanemariti “kvadratne” i više potencije epsilon), onda je

$$\frac{1}{1 + \varepsilon_y} = 1 - \varepsilon_y + \sum_{n=2}^{\infty} (-1)^n \varepsilon_y^n \approx 1 - \varepsilon_y$$

i

$$(1 + \varepsilon_x) (1 - \varepsilon_y) = 1 + \varepsilon_x - \varepsilon_y - \varepsilon_x \varepsilon_y \approx 1 + \varepsilon_x - \varepsilon_y.$$

Širenje grešaka kod dijeljenja (nastavak)

Kad to uvrstimo u prvi izraz, dobivamo

$$(x / y) (1 + \varepsilon_{/}) \approx \frac{x}{y} (1 + \varepsilon_x) (1 - \varepsilon_y) \approx \frac{x}{y} (1 + \varepsilon_x - \varepsilon_y).$$

Za relativnu grešku (približno) vrijedi

$$\varepsilon_{/} \approx \varepsilon_x - \varepsilon_y, \quad |\varepsilon_{/}| \approx |\varepsilon_x| + |\varepsilon_y|.$$

Dakle, relativne greške se oduzimaju, a ocjene zbrajaju.

U idealnom slučaju $|\varepsilon_x|, |\varepsilon_y| \leq u$, opet dobivamo približnu ocjenu relativne greške $|\varepsilon_{/}| \leq 2u$.

Vidimo da su i množenje i dijeljenje bezopasne operacije za širenje grešaka zaokruživanja.

Širenje grešaka kod zbrajanja i oduzimanja

Zbrajanje i oduzimanje. Ovdje rezultat ključno ovisi o predznacima od x i y .

Sasvim općenito, neka su x i y proizvoljnih predznaka. Za zbrajanje i oduzimanje (oznaka \pm) vrijedi

$$(x \pm y) (1 + \varepsilon_{\pm}) := [x (1 + \varepsilon_x)] \pm [y (1 + \varepsilon_y)].$$

Pogledajmo prvo trivijalne slučajeve. Ako je egzaktan rezultat $x \pm y = 0$, onda imamo dvije mogućnosti.

- Ako je $x (1 + \varepsilon_x) \pm y (1 + \varepsilon_y) = 0$, relativna greška ε_{\pm} može biti koji broj (nije određena), a prirodno je uzeti $\varepsilon_{\pm} = 0$.
- U protivnom, za $x (1 + \varepsilon_x) \pm y (1 + \varepsilon_y) \neq 0$, gornja jednakost je nemoguća, pa stavljamo $\varepsilon_{\pm} = \pm\infty$.

Širenje grešaka kod zbrajanja i oduzimanja

Pretpostavimo nadalje da je $x \pm y \neq 0$. Onda je

$$\begin{aligned}(x \pm y)(1 + \varepsilon_{\pm}) &= x(1 + \varepsilon_x) \pm y(1 + \varepsilon_y) \\ &= (x \pm y) + (x\varepsilon_x \pm y\varepsilon_y) \\ &= (x \pm y) \left(1 + \frac{x\varepsilon_x \pm y\varepsilon_y}{x \pm y} \right).\end{aligned}$$

Relativnu grešku ε_{\pm} možemo napisati u obliku **linearne kombinacije** polaznih grešaka ε_x i ε_y

$$\varepsilon_{\pm} = \frac{x\varepsilon_x \pm y\varepsilon_y}{x \pm y} = \frac{x}{x \pm y} \varepsilon_x \pm \frac{y}{x \pm y} \varepsilon_y.$$

Širenje grešaka kod zbrajanja i oduzimanja

Naravno, za nastavak rasprave **ključno** je pitanje

• koliko su **veliki faktori** uz polazne greške, tj. da li “**prigušuju**” ili “**napuhavaju**” greške.

Da ne bismo stalno pisali hrpu oznaka \pm (nepregledno), pogledajmo što se zbiva kad

• x i y imaju **isti** predznak, a

• **posebno** gledamo operacije $+$ i $-$.

Ako su x i y **različitih** predznaka, zamijenimo operaciju u suprotnu ($+$ \mapsto $-$, $-$ \mapsto $+$), pa će vrijediti isti zaključci.

Nadalje, zbrajamo i oduzimamo brojeve **istih** predznaka.

Širenje grešaka kod zbrajanja

Zbrajanje brojeva istog predznaka je bezopasno (benigno). To izlazi ovako.

Zbog istih predznaka od x i y , vrijedi $|x|, |y| \leq |x + y|$, pa je

$$\left| \frac{x}{x + y} \right|, \left| \frac{y}{x + y} \right| \leq 1.$$

To vrijedi i kad je $x = 0$ ili $y = 0$. Odavde odmah slijedi

$$|\varepsilon_+| \leq |\varepsilon_x| + |\varepsilon_y|.$$

Dakle, relativna greška se, u najgorem slučaju, zbraja.

U idealnom slučaju $|\varepsilon_x|, |\varepsilon_y| \leq u$, opet dobivamo ocjenu relativne greške $|\varepsilon_+| \leq 2u$.

Širenje grešaka kod zbrajanja (nastavak)

Uz malo truda, dobivamo i **bolju** ocjenu. Prvo uočimo da za faktore vrijedi

$$\left| \frac{x}{x+y} \right| + \left| \frac{y}{x+y} \right| = 1,$$

i još iskoristimo $|\varepsilon_x|, |\varepsilon_y| \leq \max\{|\varepsilon_x|, |\varepsilon_y|\}$. Onda je

$$\begin{aligned} |\varepsilon_+| &\leq \left| \frac{x}{x+y} \right| |\varepsilon_x| + \left| \frac{y}{x+y} \right| |\varepsilon_y| \\ &\leq \left(\left| \frac{x}{x+y} \right| + \left| \frac{y}{x+y} \right| \right) \max\{|\varepsilon_x|, |\varepsilon_y|\} \\ &= \max\{|\varepsilon_x|, |\varepsilon_y|\}. \end{aligned}$$

Širenje grešaka kod zbrajanja (nastavak)

Dakle, relativna greška zbrajanja je, u najgorem slučaju,

• **maksimum** polaznih grešaka (ne treba ih zbrajati).

U idealnom slučaju $|\varepsilon_x|, |\varepsilon_y| \leq u$, sada dobivamo ocjenu relativne greške $|\varepsilon_+| \leq u$. Bolje ne može!

Naravno, isto vrijedi i za **oduzimanje** brojeva **različitih** predznaka. I to je **bezopasno**.

Širenje grešaka kod oduzimanja

Oduzimanje brojeva istog predznaka može biti opasno, čak katastrofalno loše.

● Točnije, ne mora uvijek biti opasno, ali može!

Zašto i kada je opasno?

Zbog različitih predznaka od x i y , uz $x \neq 0$ i $y \neq 0$, sigurno vrijedi

$$|x - y| < \max\{|x|, |y|\},$$

pa je barem jedan od faktora veći od 1, tj.

$$\max\left\{\left|\frac{x}{x - y}\right|, \left|\frac{y}{x - y}\right|\right\} > 1.$$

Širenje grešaka kod oduzimanja (nastavak)

Odavde odmah slijedi da u ocjeni relativne greške

$$|\varepsilon_-| \leq \left| \frac{x}{x-y} \right| |\varepsilon_x| + \left| \frac{y}{x-y} \right| |\varepsilon_y|$$

na barem **jednom** mjestu imamo **rast** greške, a to se može dogoditi i na **oba** mjesta.

Kad je to **zaista opasno**? Ako je $|x-y| \ll |x|, |y|$, ovi faktori

$$\left| \frac{x}{x-y} \right|, \quad \left| \frac{y}{x-y} \right|,$$

mogu biti **proizvoljno veliki**, pa i relativna greška $|\varepsilon_-|$ rezultata može biti **proizvoljno velika**!

Opasno oduzimanje ili kraćenje

Opasna situacija $|x - y| \ll |x|, |y|$ znači da je

- rezultat oduzimanja brojeva istog predznaka =
- broj koji je po apsolutnoj vrijednosti mnogo manji od polaznih podataka (oba operanda),

a to znači da operandi x i y moraju biti bliski, tako da dolazi do kraćenja. Zato se ovaj fenomen obično zove

Opasno ili katastrofalno kraćenje.

Dosad smo govorili da relativna greška u tom slučaju može biti velika, ali da li se to zaista događa?

- Naime, ovdje je ipak riječ o ocjeni greške, pa se možda događa da je ocjena vrlo loša, a prava greška ipak mala!

Nažalost, nije tako! To se itekako događa u praksi!

Primjer: “Katastrofalno” kraćenje

Primjer katastrofalnog kraćenja

Zakruživanjem ulaznih podataka dolazi do male relativne greške. Kako ona može utjecati na konačni rezultat?

Primjer. Uzmimo realnu aritmetiku “računala” u bazi 10. Za mantisu (značajni dio broja) imamo $p = 4$ dekadске znamenke, a za eksponent imamo 2 znamenke (što nije bitno). Neka je

$$\begin{aligned}x &= 8.8866 = 8.8866 \times 10^0, \\y &= 8.8844 = 8.8844 \times 10^0.\end{aligned}$$

Umjesto brojeva x i y , koji nisu prikazivi, u “memoriju” spremamo brojeve $fl(x)$ i $fl(y)$, pravilno zaokružene na $p = 4$ znamenke

$$\begin{aligned}fl(x) &= 8.887 \times 10^0, \\fl(y) &= 8.884 \times 10^0.\end{aligned}$$

Primjer katastrofalnog kraćenja (nastavak)

Ovim zaokruživanjima napravili smo **malu** relativnu grešku u x i y (ovdje je $u = \frac{1}{2} b^{-p} = 5 \times 10^{-5}$).

Razliku $fl(x) - fl(y)$ računamo tako da **izjednačimo eksponente** (što već jesu), **oduzmemo** značajne dijelove (mantise), pa **normaliziramo**

$$\begin{aligned} fl(x) - fl(y) &= 8.887 \times 10^0 - 8.884 \times 10^0 \\ &= 0.003 \times 10^0 = 3.??? \times 10^{-3}. \end{aligned}$$

Kod normalizacije, zbog pomaka “**ulijevo**”, pojavljuju se

● **?** = znamenke koje više **ne možemo** restaurirati (ta informacija se **izgubila** — zaokruživanjem x i y).

Što sad?

Primjer katastrofalnog kraćenja (nastavak)

Računalo radi **isto** što bismo i mi napravili:

👉 na ta mjesta ? upisuje 0.

Razlog: da rezultat bude **točan**, ako su **polazni** operandi **točni**. Dakle, ovo oduzimanje je **egzaktno** i u aritmetici računala.

Konačni **izračunati** rezultat je $fl(x) - fl(y) = 3.000 \times 10^{-3}$.

Pravi rezultat je

$$\begin{aligned}x - y &= 8.8866 \times 10^0 - 8.8844 \times 10^0 \\ &= 0.0022 \times 10^0 = 2.2 \times 10^{-3}.\end{aligned}$$

Već **prva** značajna znamenka u $fl(x) - fl(y)$ je **pogrešna**, a relativna greška je **ogromna**! Uočite da je ta znamenka (**3**), ujedno, i **jedina** koja nam je ostala — sve ostalo se **skratilo**!

Primjer katastrofalnog kraćenja (nastavak)

Prava **katastrofa** se događa ako $3.??? \times 10^{-3}$ uđe u naredna zbrajanja (oduzimanja), a onda se **skrati** i ta **trojka!**

Uočite da je **oduzimanje** $fl(x) - fl(y)$ bilo **egzaktno** i u aritmetici našeg “**računala**”, ali **rezultat je**, svejedno, **pogrešan**.

Krivac, očito, **nije** **oduzimanje** (kad je egzaktno).

- Uzrok su **polazne greške** u operandima $fl(x)$, $fl(y)$.

Ako njih **nema**, tj. ako su polazni operandi **egzaktni**,

- i dalje, naravno, dolazi do **kraćenja**,

- ali je **rezultat** (uglavnom, a po IEEE standardu **sigurno**) **egzaktan**,

pa se ovo kraćenje onda zove **benigno kraćenje**.

Ponavljjanje i dodatak

Ponavljanje Prog1 i dodatak

Za ponavljanje prikaza brojeva u računalu, pogledajte

- 3. i 4. predavanje iz Programiranja 1.

Pogledajte i dodatak ovom predavanju. Sadrži

- ponavljanje gradiva iz Prog1 o prikazu realnih brojeva u računalu i greškama zaokruživanja,
- još poneke stvari o širenju grešaka prilikom aritmetičkih operacija.

Primjeri “grešaka” iz prakse

Promašaj raketa Patriot

U prvom Zaljevskom ratu, 25. veljače 1991. godine, američke rakete Patriot nisu uspjele oboriti iračku Scud raketu iznad Dhahrana u Saudijskoj Arabiji.

- Scud raketa je pukim slučajem pala na američku vojnu bazu — usmrativši 28 i ranivši stotinjak ljudi.



Promašaj raketa Patriot (nastavak)

Istraga otkriva sljedeće:

- Računalo koje je upravljalo Patriot raketama, vrijeme je brojilo u desetinkama sekunde proteklim od trenutka paljenja (uključivanja) sustava.
- Desetinka sekunde binarno

$$0.1_{10} = (0.00011)_2.$$

- To računalo prikazivalo je realne brojeve korištenjem nenormalizirane mantise duljine 23 bita.
- Spremanjem broja 0.1 u registar takvog računala radi se (apsolutna) greška $\approx 9.5 \cdot 10^{-8}$ (sekundi).

Ne izgleda puno . . . , a kamo li opasno.

Promašaj raketa Patriot (nastavak)

Detalji:

- Računalo je bilo u pogonu 100 sati, pa je ukupna greška zaokruživanja bila (stalno se zbraja, svakih 0.1 sekundi)

$$100 \cdot 60 \cdot 60 \cdot 10 \cdot 9.5 \cdot 10^{-8} = 0.34 \text{ s.}$$

- Scud raketa putuje brzinom $\approx 1.6 \text{ km/s}$, pa je “tražena” više od pola kilometra daleko od stvarnog položaja.
- Greška je uočena dva tjedna ranije, nakon 8 sati rada jednog drugog sustava. Modifikacija programa stigla je dan nakon nesreće.
- Posade sustava mogle su i dva tjedna ranije dobiti uputu “isključi/uključi računalo” svakih nekoliko sati — ali je nisu dobile.

Samouništenje Ariane 5

Raketa *Ariane 5* lansirana 4. lipnja 1995. godine iz Kouroua (Francuska Gvajana).

- Nosila je u putanju oko Zemlje komunikacijske satelite vrijedne 500 milijuna USD.
- 37 sekundi nakon lansiranja izvršila je samouništenje.



Samouništenje Ariane 5 (nastavak)

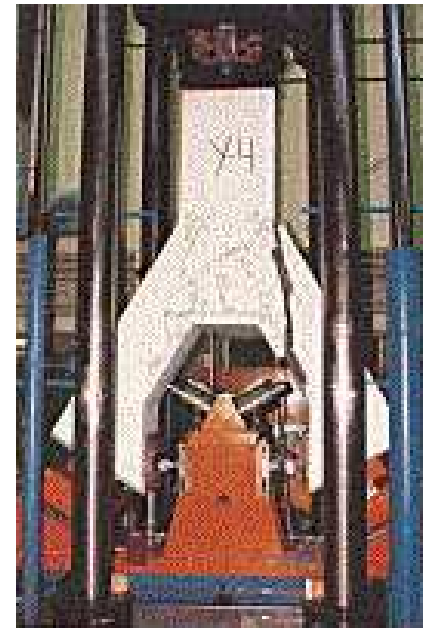
Objašnjenje:

- U programu za vođenje rakete postojala je varijabla koja je registrirala (pamtila) horizontalnu brzinu rakete (stvarno, nije koristila ničemu).
- Greška je nastupila kad je program pokušao pretvoriti
 - preveliki 64-bitni realni broj
 - u 16-bitni cijeli broj.
- Računalo je javilo grešku, što je izazvalo samouništenje.
- Isti program bio je korišten u prijašnjoj sporijoj verziji Ariane 4, pa do katastrofe nije došlo.

Potonuće naftne platforme

Naftna platforma **Sleipner A** **potonula** je prilikom prvog sidrenja, 23. kolovoza 1991. godine u blizini Stavangera.

- Baza platforme su **24** betonske ćelije, od kojih su **4** produljene u šuplje stupove na kojima leži paluba.



Potonuće naftne platforme (nastavak)

Razlozi nesreće:

- Prilikom uronjavanja baze došlo je do **pucanja veza** među ćelijama (v. desnu sliku).
- Rušenje na dno mora je izazvalo **potres** jačine **3.0** stupnja po Richterovoj ljestvici i **štetu** od **700** milijuna USD.
- Greška je nastala u **projektiranju**, primjenom **standardnog paketa** programa, kad je upotrijebljena metoda konačnih elemenata s **nedovoljnom točnošću**.
- Proračun je dao naprezanja **47%** **manja** od stvarnih.
- **Točnijim** proračunom utvrđeno je da su ćelije **morale** popustiti na dubini od **62** metra, a popustile su na **65** metara!