Može li se vjerovati računalu?
Računalo je izračunalo ...

Saša Singer

PMF – Matematički odjel,
Sveučilište u Zagrebu

e–mail: singer@math.hr

Sanja Singer

Katedra za matematiku i nacrtnu geometriju, FSB,
Sveučilište u Zagrebu

e–mail: ssinger@math.hr

Matematički klub, XV. Gimnazija

Zagreb, 15. travnja 2002.

Računalo je izračunalo ...

ili

"Kompjuter je tak izbacil ..."

Tako stvari počinju. A kako mogu završiti, vidjet ćete na kraju!

Za početak, da li ste već čuli nešto takvo?

To su tipični predstavnici modernog pogleda na računala i rezultate njihovog rada!

Ako je sve ispalo dobro — zaslužno je računalo. Ono je

- nadnaravno sposobno,
- samostalno misli,
- i, naravno, bezgrešno!

Ako je nešto ispalo loše — opet, krivac je računalo. Ono je

- svojeglavo,
- ne reagira "na podražaje"
- ili, potpuno poludilo!

Treba li tome vjerovati? I tko je zaslužan ili krivac?

Računalo ili ljudi "oko" njega (programeri, korisnici)?

Računalo je izračunalo . . .

Idemo pogledati prvobitnu i osnovnu primjenu računala

- računanje (engl. computing)

i pokušati odgovoriti na pitanje:

- Treba li vjerovati računalu?

Što radi računalo kad računa?

- Izvršava neki program,
- Na kraju, daje neke rezultate.

Dakle, treba li vjerovati dobivenim rezultatima?

Vjerovati? NE! Nikad!

Skrivanje iza računala (Računalo je izračunalo . . .)
=
jadno pokriće nečije nesposobnosti.

Kad to čujemo, posebno od inženjera i znanstvenika,

- hvata nas strah!

Zašto?

To je dokaz da dobivene rezultate nitko nije pogledao, nego da im se slijepo vjeruje.

Što je loše u tome?

Zapravo, samo slijepo vjerovanje. Ako netko zna zašto vjeruje — svaka čast, ali baš to nije jednostavno (v. dalje)!

Slijepo vjerovanje — ili nekoliko fama o računalima:

- računalom se sve može izračunati;
- rješenje se uvijek dobiva u kratkom vremenu;
- računalo uvijek daje točne rezultate.

Međutim (nažalost?):

Ništa od toga nije istinito!

Za prve dvije stvari — to je (valjda) očito (iako bismo i tu imali što pričati).

Pogledajmo zadnju — točnost.

Dobiveni rezultati mogu biti pogrešni iz razno-raznih razloga, a najčešći krivac nije računalo.

Greške!

Grubo govoreći, imamo dvije vrste grešaka:

- nepotrebne (mogu se izbjeći);
- nužne ili očekivane.

Nepotrebno (uglavnom) griješe ljudi:

- greške u programiranju;
- pogrešna (nekritička) primjena programa.

Računala — vrlo rijetko (bar u novije vrijeme):

- greška dijeljenja u jednoj seriji Pentium procesora 1994. g.

Ranije — princip “demokracije” ili “nadglasavanja”.

Već iz ovog slijedi

Pouka: Rezultate treba provjeriti!

Nažalost, u to je ljude najteže uvjeriti.

- U praksi, brojevi uvijek imaju neko značenje!

Zamislite temperaturu vode od 120°C!

Ove nepotrebne greške su razumljive ili “očekivane” — po sistemu: “tko radi, taj i griješi”.

Ali, što su ostale greške — i zašto su nužne u praksi?

Odakle "nužne" ili neizbježne greške?

Izvori grešaka kod rješavanja praktičnih problema su:

- model,
- metoda za rješavanje modela,
- ulazni podaci (mjerenja),
- aritmetika računala.

Dakle, izvora grešaka je puno – javljaju se na svakom koraku. Oprez je nužan.

Prve 3 stvari, uglavnom, ne ovise o računalu, ali zadnja da!

Stvarnost je još malo grublja:

- računalo skoro uvijek daje pogrešne rezultate.

Prirodno pitanje:

- Može li greška aritmetike biti dominantna u odnosu na ostale, tako da je rezultat zbog nje besmislen?

Opet, nažalost

MOŽE!

I u to je ljude teško uvjeriti, čak i kad vide sve primjere koji dolaze.

Slijepo vjerovanje rezultatima može biti pogibeljno!

Prikaz brojeva i aritmetika računala

Što zaista moramo znati o aritmetici računala da bismo imali elementarnu podlogu za vjerovanje rezultatima?

- Prikaz brojeva u računalu;
- Porijeklo grešaka zaokruživanja;
- Osnovna pravila za greške (jer one nisu slučajne).

Prikaz brojeva

U računalu postoje dva bitno različita tipa (skupa) brojeva:

- "cijeli" brojevi — `integer`,
- "realni" brojevi — `real`.

Osnovna razlika od "matematičkih" skupova $\mathbb{Z}$ i $\mathbb{R}$:

- oba skupa su konačni skupovi,

dakle, pravi podskupovi od $\mathbb{Z}$, odnosno $\mathbb{R}$.

Prva posljedica: U oba tipa postoje brojevi koje ne možemo prikazati u računalu!

Zajedničko svojstvo:

- Prikaz brojeva koristi pozicioni zapis u bazi 2.

Cijeli brojevi — `integer`

Osnovna svojstva prikaza i aritmetike:

- duljina n bitova, n fiksan,
- aritmetika — modularna aritmetika u prstenu ostataka modulo 2^n,
- samo je sistem ostataka simetričan oko 0, (tako da je "prvi bit predznak").

Prikazivi brojevi su:

$$-2^{n-1}, \ldots, -1, 0, 1, \ldots, 2^{n-1} - 1.$$

Tipične vrijednosti za n su: 16, 32 i 64.

Oprez: aritmetika cijelih brojeva je zaista modularna i računalo standardno ne javlja grešku kod "zatvaranja kruga" pri izvršavanju aritmetičkih operacija.

Primjer za $n = 32$:

$$\begin{aligned} \text{najveći broj} &= \quad 2^{31} - 1 = \quad 2\,147\,483\,647 \\ \text{najmanji broj} &= -2^{31} \qquad\;\; = -2\,147\,483\,648 \end{aligned}$$

pa je

$$(2^{31} - 1) + 1 = -2^{31}.$$

To su jedine "čarolije" cjelobrojne aritmetike!

Realni brojevi — `real`

Realni brojevi r se prikazuju u obliku (znanstvena notacija):

$$r = \pm m \cdot 2^e,$$

gdje je

- eksponent $e =$ cijeli broj u određenom rasponu,
- mantisa $m =$ racionalni broj za koji je $1/2 \leq m < 1$ (tj. mantisa započinje s $0.1\ldots$).
- Dogovor: za $r = 0$, mantisa je 0 i $e = 0$.

Osim toga, zbog konačnosti prikaza:

- eksponent e je s-bitni cijeli broj,
- za mantisu m pamti se prvih t znamenki iza binarne točke.

To izgleda ovako:

mantisa

$\pm$	m_{-1}	m_{-2}	$\cdots$	m_{-t}

eksponent

$\pm$	e_{s-2}	e_{s-3}	$\cdots$	e_0

Skup realnih brojeva prikazivih u računalu parametriziran je duljinom mantise t i duljinom eksponenta s, u oznaci

$$\mathbb{R}(t, s).$$

Zaokruživanje

Postoje realni brojevi koje ne možemo egzaktno spremiti u računalo, čak i kad su unutar prikazivog raspona brojeva.

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

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

Ako mantisa od x ima više od t znamenki, sprema se (najbliža) aproksimacija $fl(x) \in \mathbb{R}(t, s)$ koja se može prikazati kao

$$fl(x) = \pm \left(\sum_{k=1}^{t} b^*_{-k} 2^{-k} \right) 2^{e^*}.$$

Broj $fl(x)$ dobivamo zaokruživanjem broja x:

- prva odbačena znamenka 1 — zaokruži nagore,
- prva odbačena znamenka 0 — zaokruži nadolje.

Time smo napravili grešku zaokruživanja $\leq \frac{1}{2}$ "zadnjeg bita":

$$\text{apsolutna greška} \leq 2^{-t-1+e}.$$

Zgodnije je ocijeniti relativnu grešku

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

Dakle, imamo vrlo malu relativnu grešku.

Oznaka: jedinična greška zaokruživanja (engl. unit roundoff)

$$u := 2^{-t}.$$

Zaokruživanje (nastavak)

Dakle, ako je $x \in \mathbb{R}$ unutar raspona brojeva prikazivih u računalu, onda se umjesto x sprema zaokruženi broj $fl(x) \in \mathbb{R}(t, s)$ i vrijedi

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

gdje je ε relativna greška napravljena tim zaokruživanjem.

IEEE standard za realne tipove:

	`single`	`double`	`extended`
duljina	32 bita	64 bita	80 bitova
mantisa	$23 + 1$ bit	$52 + 1$ bit	64 bita
eksponent	8 bitova	11 bitova	15 bitova
u	2^{-24}	2^{-53}	2^{-64}
$u \approx$	$5.96 \cdot 10^{-8}$	$1.11 \cdot 10^{-16}$	$5.42 \cdot 10^{-20}$
raspon $\approx$	$10^{\pm 38}$	$10^{\pm 308}$	$10^{\pm 4932}$

Ima još nešto "čarolija" u prikazu, ali one nisu bitne ("skriveni bit", posebni eksponenti, ...).

Napomena: svako čitanje podataka rezultira nekom greškom zaokruživanja!

Zaokruživanje u aritmetici

Aritmetičke operacije se mogu izvesti samo na operandima koji su već spremljeni u memoriji, dakle pripadaju skupu $\mathbb{R}(t, s)$.

Osnovna pretpostavka: za sve četiri osnovne aritmetičke operacije vrijedi ista ocjena greške zaokruživanja kao i za prikaz brojeva.

Preciznije: Neka $\circ$ označava bilo koju operaciju $+$, $-$, $*$, $/$. Onda za $x, y \in \mathbb{R}(t, s)$ vrijedi

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

za sve $x, y \in \mathbb{R}(t, s)$ za koje je $x \circ y$ u dozvoljenom rasponu.

Napomena: oznaka $fl(izraz)$ označava izračunatu vrijednost izraza (mora biti prikaziva).

Ova ocjena je ekvivalentna idealnom izvođenju operacija:

- egzaktno izračunaj rezultat operacije $x \circ y$,
- zaokruži ga, pri spremanju rezultata u memoriju!

Vidimo da svaki izračunati rezultat ima neku grešku. Osim toga, zaokruživanje se vrši nakon svake pojedine operacije!

Kad imamo puno aritmetičkih operacija, dolazi do

akumulacije grešaka zaokruživanja.

Širenje grešaka zaokruživanja

Ključno pitanje: Možemo li išta reći o tom "rasprostiranju" grešaka zaokruživanja?

Možemo!

Nažalost, aritmetika računala nije egzaktna i u njoj ne vrijede uobičajeni zakoni za operacije. Na primjer, za zbrajanje i množenje nema asocijativnosti, distributivnosti.

Međutim, analiza pojedinih operacija postaje bitno lakša, ako uočimo:

- Greške zaokruživanja možemo interpretirati i kao egzaktne operacije, ali na "malo" pogrešnim podacima!

Dovoljno je $1 + \varepsilon$ u ocjeni greške "zalijepiti" za x i/ili y. To je isto kao da operand(i) ima(ju) grešku na ulazu u operaciju, a operacija $\circ$ je egzaktna. Onda možemo koristiti "normalna" pravila aritmetike za analizu grešaka.

Pretpostavimo onda da su podaci x i y malo perturbirani, s relativnim greškama

$$|\varepsilon_x|, |\varepsilon_y| \leq u.$$

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

Množenje i dijeljenje

Množenje: benigno

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

uz ocjenu relativne greške

$$|\varepsilon_*| \leq 2u.$$

Greška se samo zbraja.

Dijeljenje: benigno

$$\begin{aligned} \frac{x\,(1+\varepsilon_x)}{y\,(1+\varepsilon_y)} &\approx \frac{x}{y}\,(1+\varepsilon_x)\,(1-\varepsilon_y) \\ &:= \frac{x}{y}\,(1+\varepsilon_/), \end{aligned}$$

uz istu ocjenu relativne greške

$$|\varepsilon_/| \leq 2u.$$

Greška se opet samo zbraja.

Dakle, množenje i dijeljenje su bezopasne operacije za širenje grešaka zaokruživanja.

Zbrajanje i oduzimanje

Općenito:

Neka su x i y proizvoljnog predznaka. Za zbrajanje (oduzimanje) vrijedi:

$$x\,(1+\varepsilon_x) + y\,(1+\varepsilon_y) = (x+y)\Big(1 + \frac{x\varepsilon_x + y\varepsilon_y}{x+y}\Big).$$

Ako je $x+y \neq 0$, definiramo

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

Isti predznak ("zbrajanje"): benigno

Tada vrijede ocjene

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

pa je $|\varepsilon_\pm| \leq 2u$. Greška se opet samo zbraja.

Različiti predznak ("oduzimanje"): opasno

Ako je $|x+y| \ll |x|, |y|$, kvocijenti

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

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

Opasno oduzimanje

OPASNOST — rezultat zbrajanja brojeva suprotnog predznaka = broj koji je po apsolutnoj vrijednosti mnogo manji od polaznih podataka, tzv.

Opasno kraćenje.

Primjer — na "računalu" u bazi 10. Za mantisu imamo $t = 4$ dekadske znamenke, a za eksponent $s = 2$ znamenke. Uzmimo

$$x = 0.88866 = 0.88866 \cdot 10^0, \quad y = 0.88844 = 0.88844 \cdot 10^0.$$

Umjesto brojeva x i y, spremili smo

$$fl(x) = 0.8887 \cdot 10^0, \quad fl(y) = 0.8884 \cdot 10^0$$

i napravili malu relativnu grešku. Oduzimamo znamenku po znamenku mantise, pa normaliziramo

$$0.8887 \cdot 10^0 - 0.8884 \cdot 10^0 = 0.0003 \cdot 10^0 = 0.3??? \cdot 10^{-3}.$$

? — znamenke koje više ne možemo restaurirati, pa računalo na ta mjesta upisuje 0. Pravi rezultat je $0.22 \cdot 10^{-3}$ — prva značajna znamenka pogrešna!

Oduzimanje je bilo egzaktno za $fl(x)$ i $fl(y)$, ali rezultat je pogrešan.

Katastrofa — ako $0.3??? \cdot 10^{-3}$ uđe u naredna zbrajanja i/ili oduzimanja i ako se pritom "skrati" i ta trojka.

Kvadratna jednadžba

Uzmimo da treba riješiti (realnu) kvadratnu jednadžbu

$$ax^2 + bx + c = 0,$$

gdje su a, b i c zadani i $a \neq 0$.

Matematički gledano, problem je trivijalan: imamo 2 rješenja

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Numerički gledano, problem je mnogo izazovniji:

- ni uspješno računanje po ovoj formuli,
- ni točnost izračunatih korijena,

ne možemo uzeti "zdravo za gotovo".

Primjer: $x^2 - 56x + 1 = 0$. U aritmetici s 5 decimala dobijemo

$$x_1 = \frac{56 - \sqrt{3132}}{2} = \frac{56 - 55.964}{2} = 0.018000,$$
$$x_2 = \frac{56 + \sqrt{3132}}{2} = \frac{56 + 55.964}{2} = 55.982.$$

Točna rješenja su

$$x_1 = 0.0178628\ldots \quad \text{i} \quad x_2 = 55.982137\ldots.$$

Manji od ova dva korijena ima samo dvije točne znamenke (kraćenje).

Kvadratna jednadžba — popravak

Prvo izračunamo većeg po apsolutnoj vrijednosti, po formuli

$$x_2 = \frac{-(b + \mathrm{sign}(b)\sqrt{b^2 - 4ac})}{2a},$$

a manjeg po apsolutnoj vrijednosti, izračunamo iz

$$x_1 \cdot x_2 = \frac{c}{a}$$

(Vieta), tj.

$$x_1 = \frac{c}{x_2\, a}.$$

Opasnog kraćenja za x_1 više nema!

Primjer 1.

Vrijednost

$$f_n(x) = (x - n)^{10}, \quad n = 0, \ldots, 10,$$

računamo u aritmetici računala u okolini točke n.

Primijetite da je graf funkcije $(x - n)^{10}$ translatirani graf funkcije x^{10} za n jedinica udesno.

Funkcijsku vrijednost funkcija f_n možemo izračunati na više načina koji su matematički ekvivalentni, ali nisu numerički jednaki:

- translacijom grafa funkcije x^{10} za n jedinica udesno,
- korištenjem binomne formule

$$(x - n)^{10} = \sum_{k=0}^{10} \binom{10}{k} x^k (-n)^{10-k},$$

  s tim da polinom na desnoj strani računamo Hornerovom shemom.

Što je točnije?

Odgovor: U okolini točke n je $(x-n)^{10}$ mali broj. Pogledajmo kakvi su članovi u sumi na desnoj strani. Koeficijenti s desne strane su alternirajući po predznaku i rastu s porastom n.

U sumi mora doći do kraćenja, pa je rezultat bolje izračunati direktno.

Kako izgledaju grafovi?

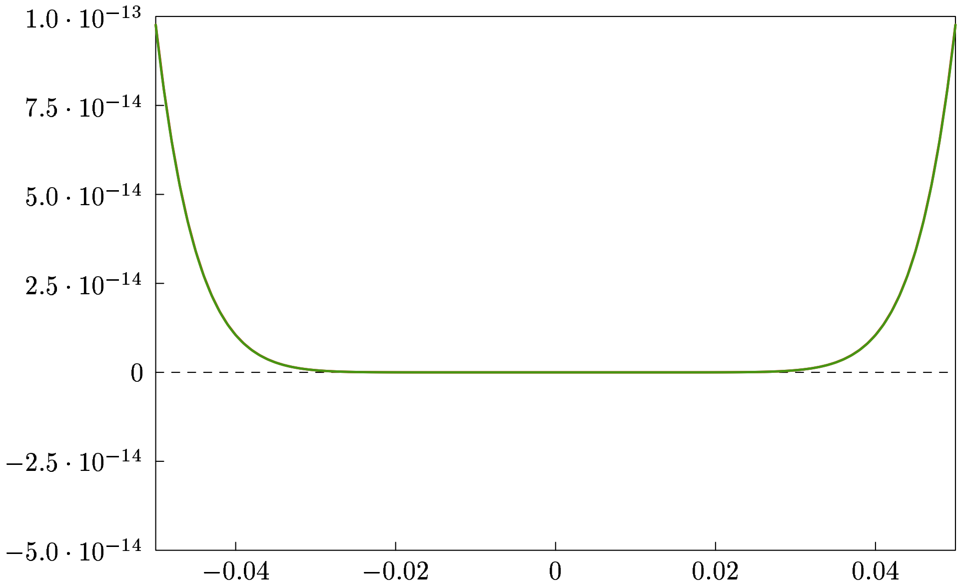
- zeleno — graf dobiven translacijom,
- crveno — korištenjem binomne formule.

Za svaki n crtamo dvije slike grafa funkcije f_n:

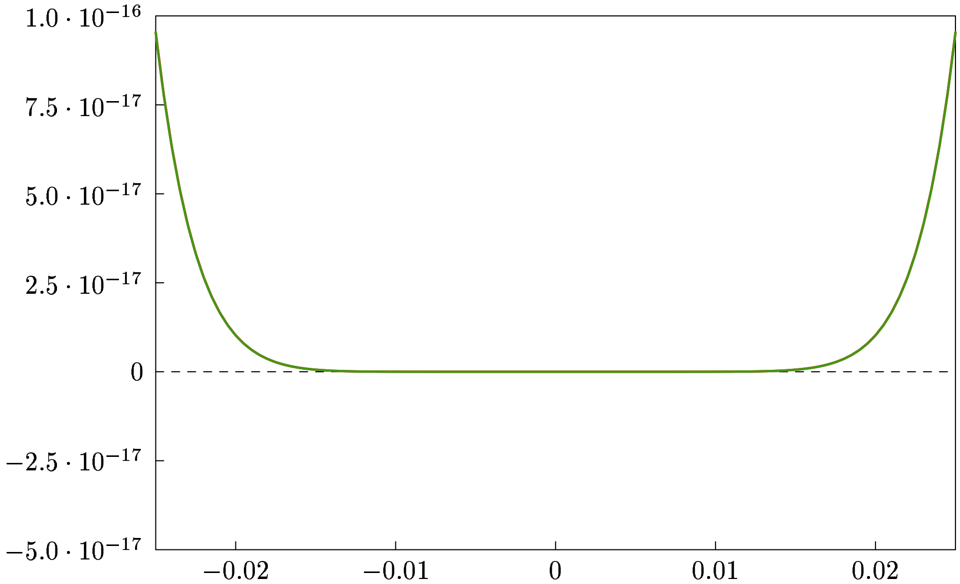
- na intervalu $[n - 0.05, n + 0.05]$,
- na intervalu $[n - r, n + r]$, gdje je r odabran tako da ovaj interval sadrži numeričke nultočke od f_n.

Obratite pažnju na skale po x i y!

$$(x-0)^{10} = x^{10}$$

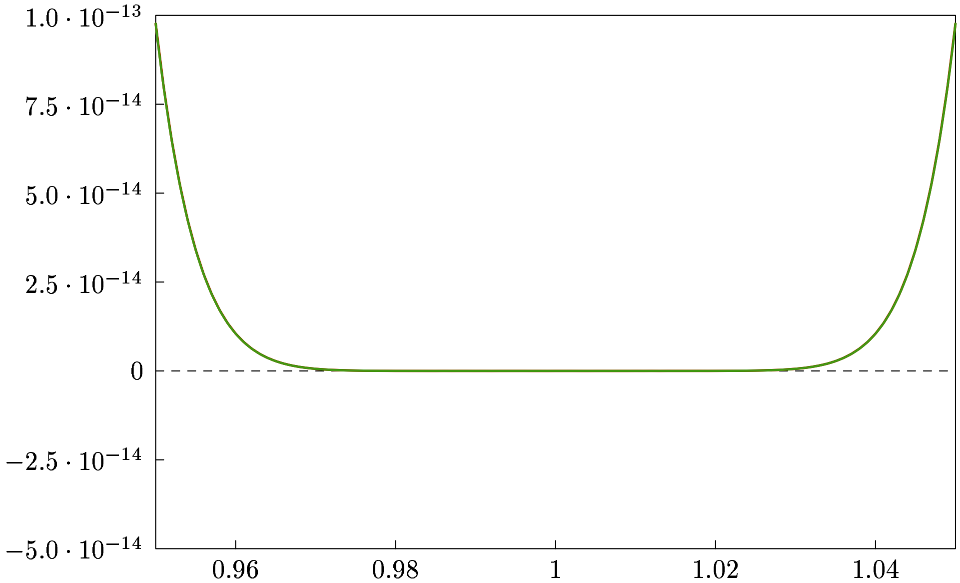


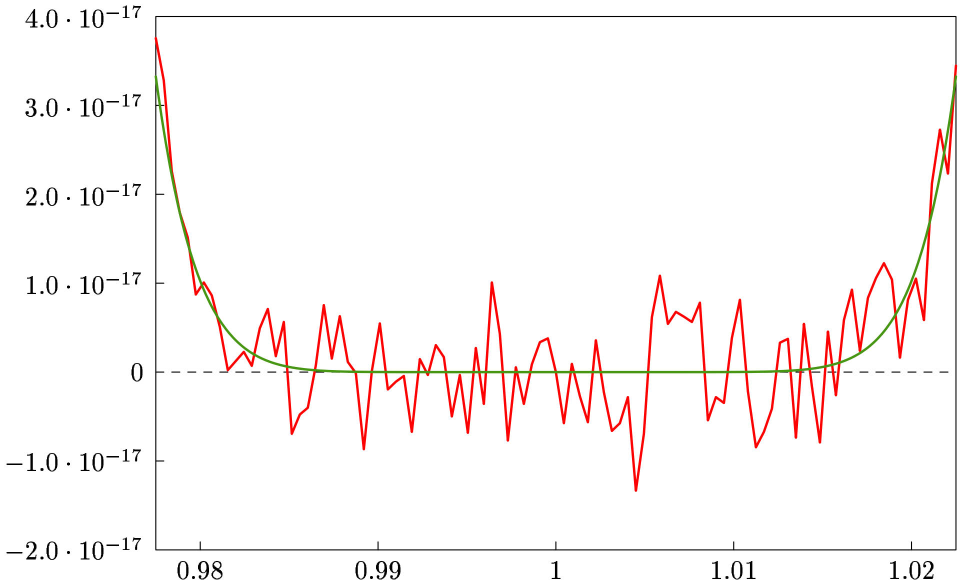
$1.0 \cdot 10^{-13}$
$7.5 \cdot 10^{-14}$
$5.0 \cdot 10^{-14}$
$2.5 \cdot 10^{-14}$
0
$-2.5 \cdot 10^{-14}$
$-5.0 \cdot 10^{-14}$
-0.04
-0.02
0
0.02
0.04



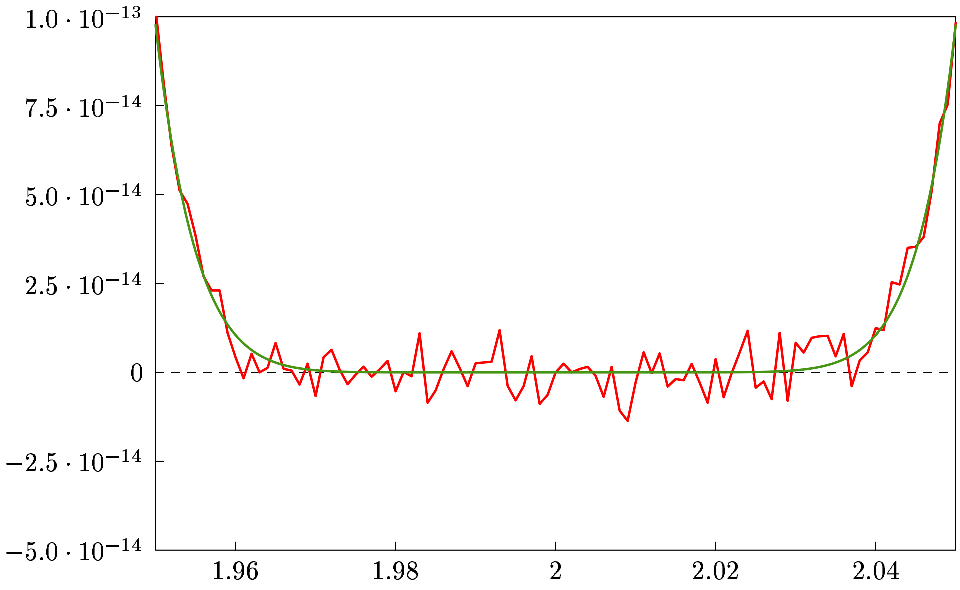
$1.0 \cdot 10^{-16}$
$7.5 \cdot 10^{-17}$
$5.0 \cdot 10^{-17}$
$2.5 \cdot 10^{-17}$
0
$-2.5 \cdot 10^{-17}$
$-5.0 \cdot 10^{-17}$
-0.02
-0.01
0
0.01
0.02

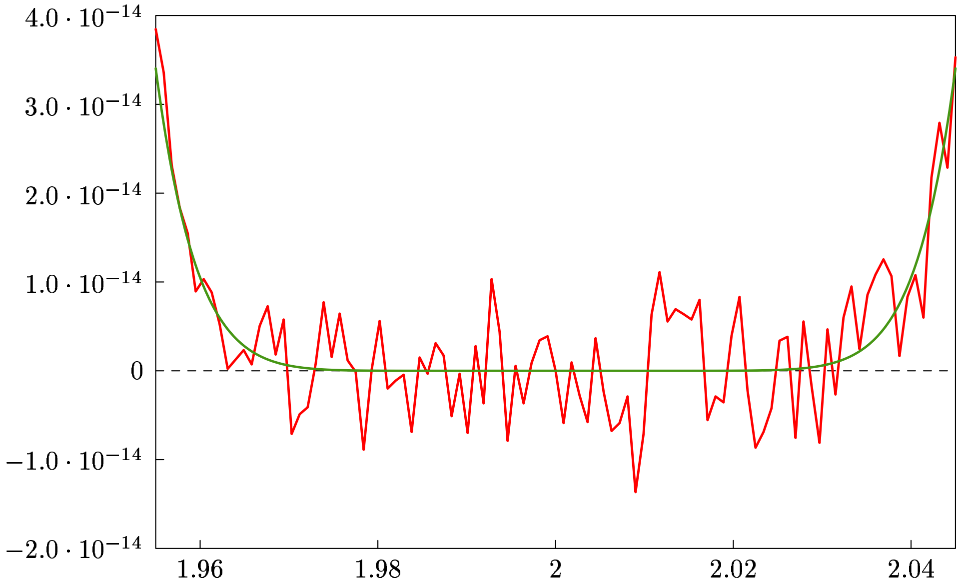
$$(x-1)^{10} = x^{10} - 10x^9 + 45x^8 - 120x^7 + 210x^6 - 252x^5 + 210x^4 - 120x^3 + 45x^2 - 10x + 1$$



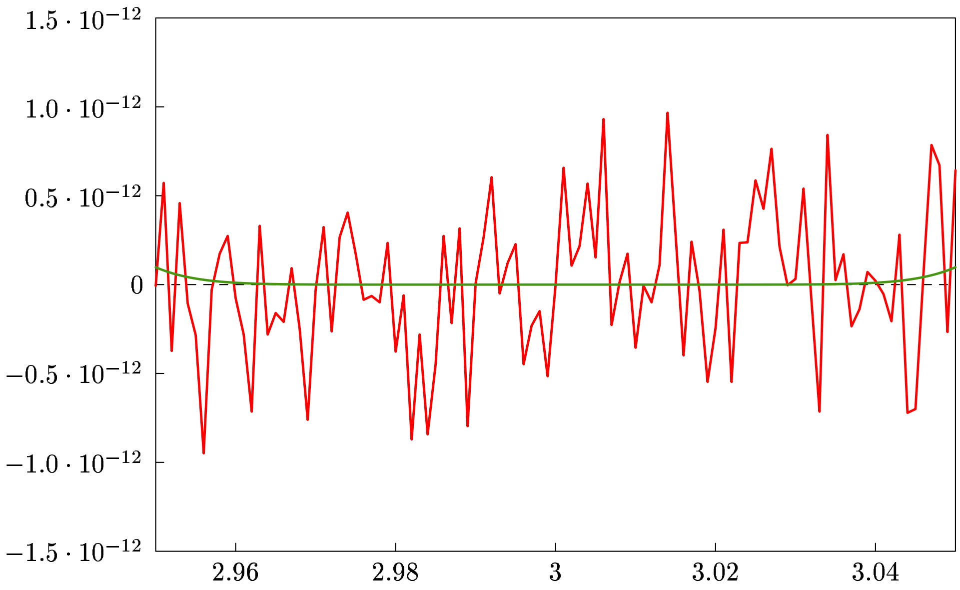


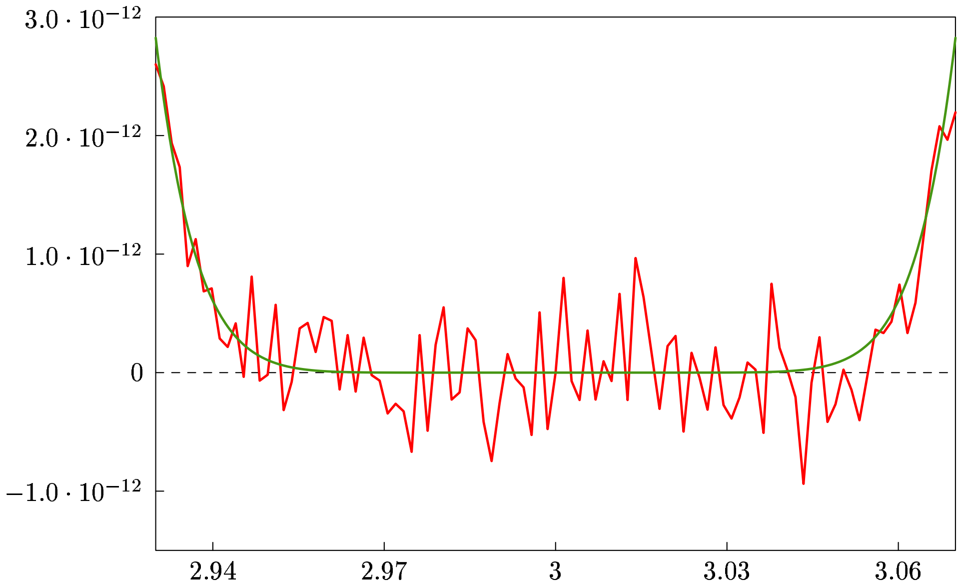
$$\begin{aligned}(x-2)^{10} = x^{10} - 20x^9 + 180x^8 - 960x^7 + 3360x^6 \\ - 8064x^5 + 13440x^4 - 15360x^3 \\ + 11520x^2 - 5120x + 1024\end{aligned}$$



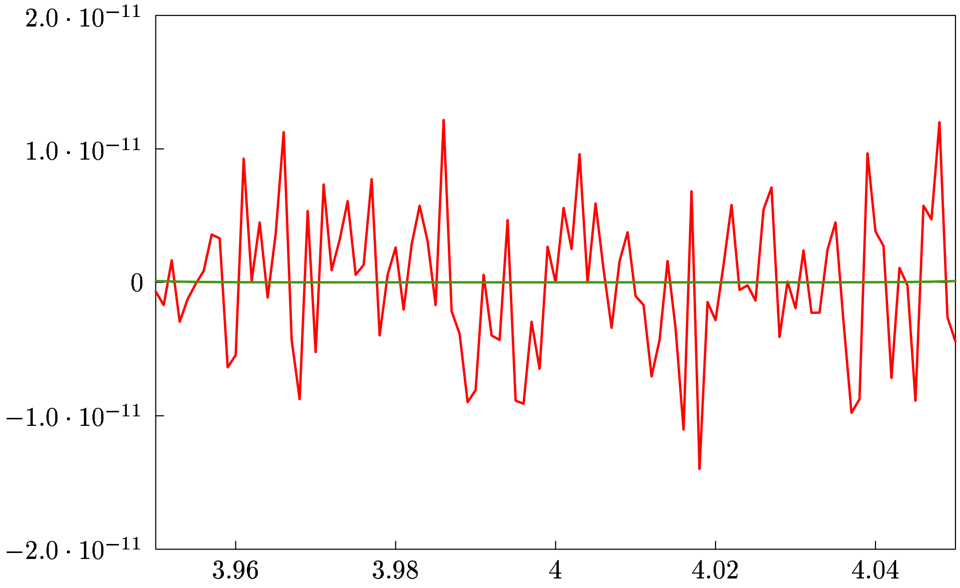


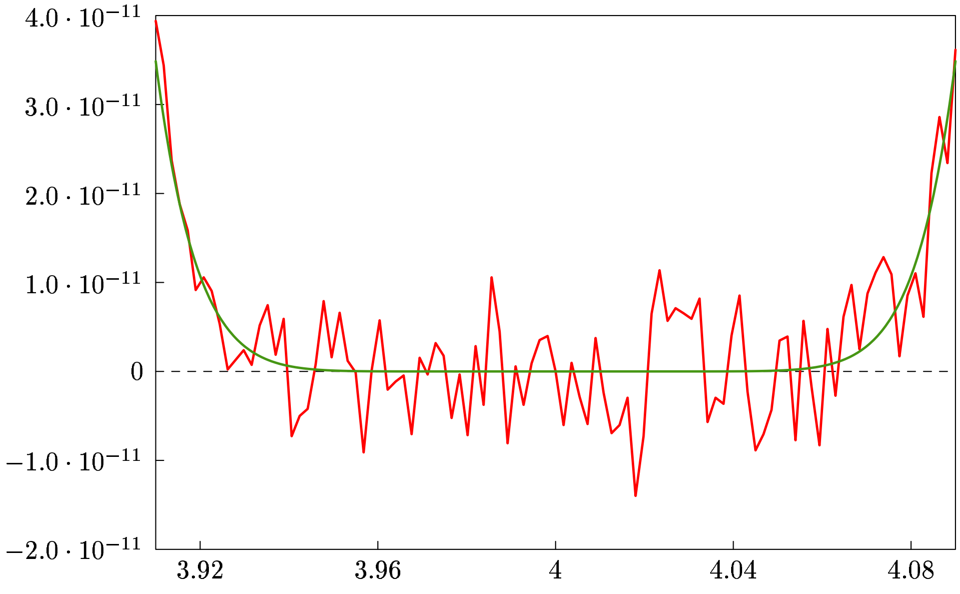
$$(x-3)^{10} = x^{10} - 30x^9 + 405x^8 - 3240x^7 + 17010x^6 - 61236x^5 + 153090x^4 - 262440x^3 + 295245x^2 - 196830x + 59049$$



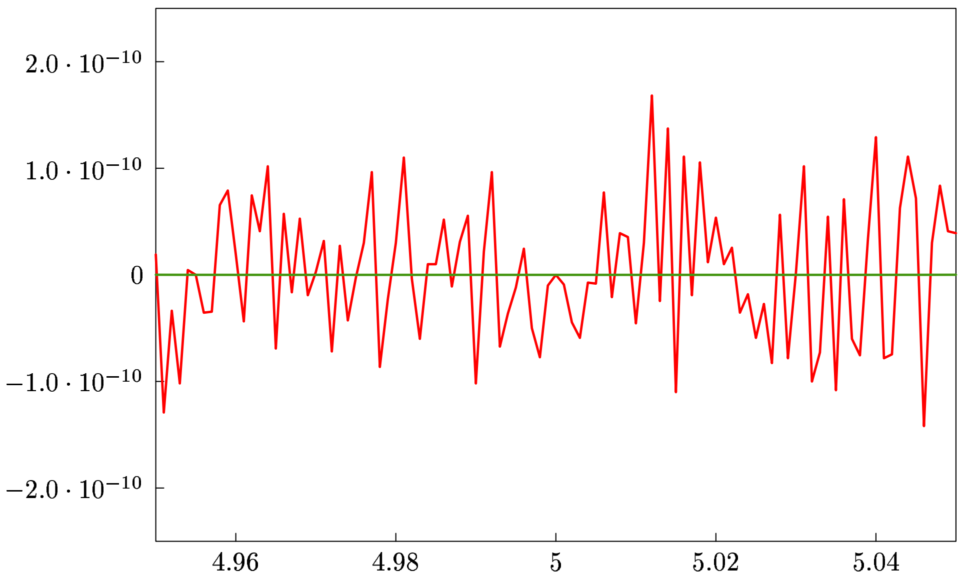


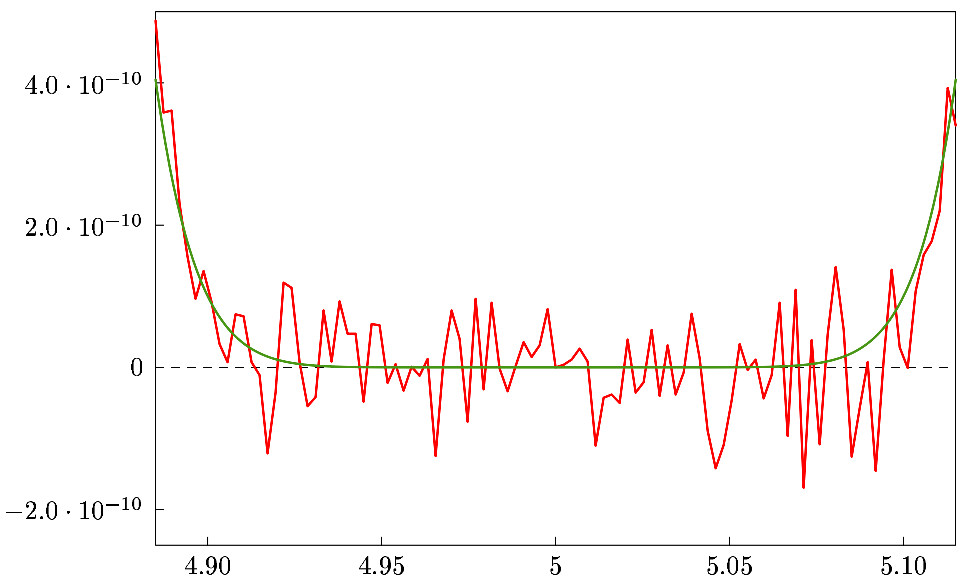
$$(x-4)^{10} = x^{10} - 40x^9 + 720x^8 - 7680x^7 + 53760x^6 - 258048x^5 + 860160x^4 - 1966080x^3 + 2949120x^2 - 2621440x + 1048576$$



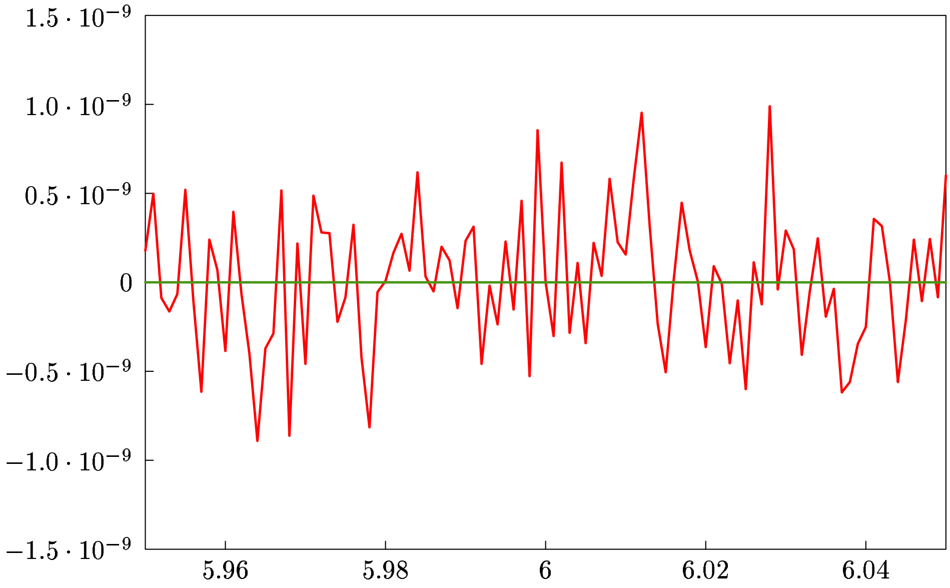


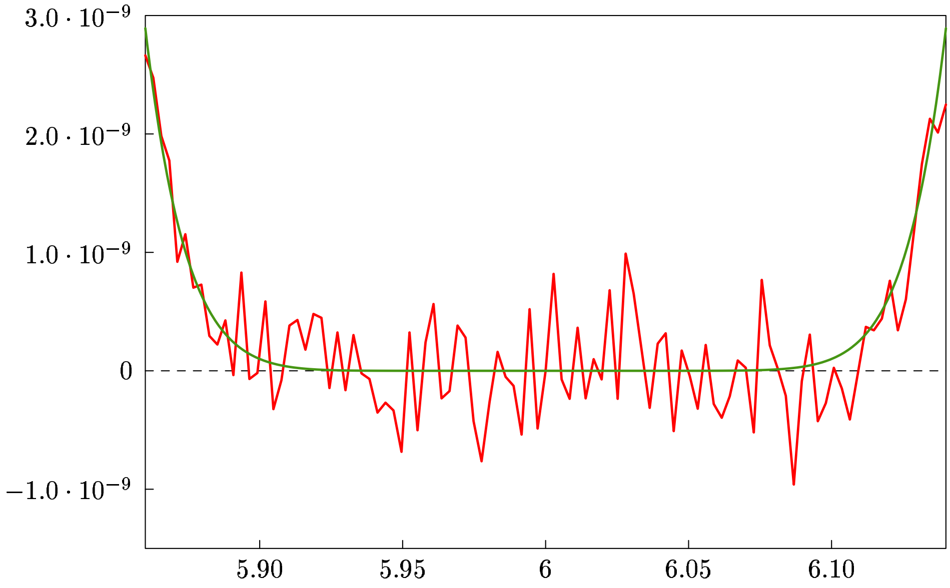
$$
\begin{aligned}
(x-5)^{10} = x^{10} &- 50x^9 + 1125x^8 - 15000x^7 + 131250x^6 \\
&- 787500x^5 + 3281250x^4 - 9375000x^3 \\
&+ 17578125x^2 - 19531250x + 9765625
\end{aligned}
$$



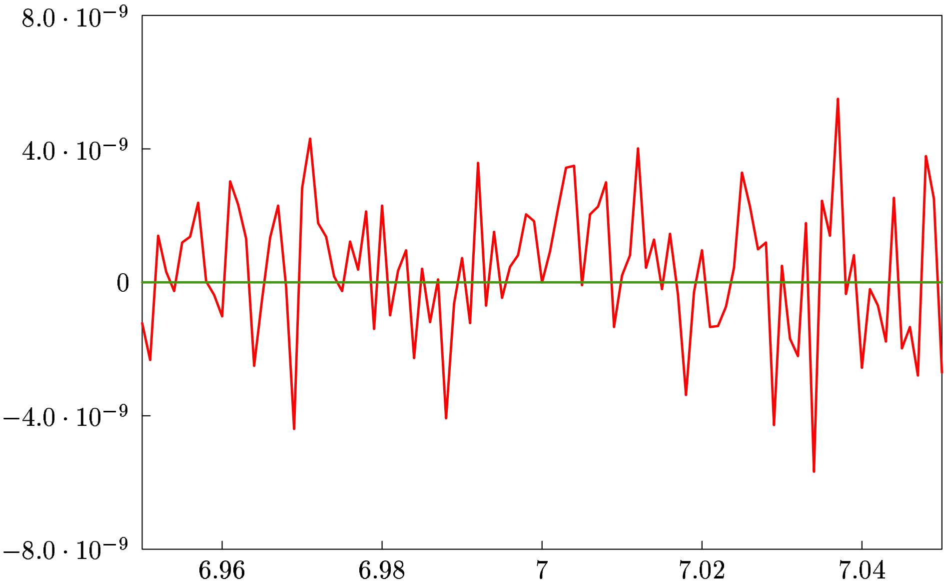


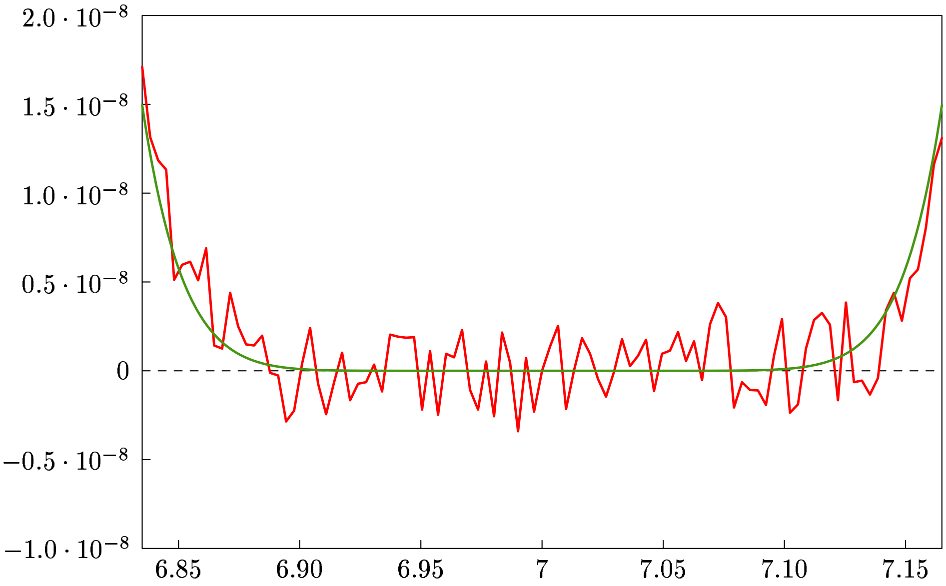
$$
\begin{aligned}
(x-6)^{10} = x^{10} &- 60x^9 + 1620x^8 - 25920x^7 + 272160x^6 \\
&- 1959552x^5 + 9797760x^4 - 33592320x^3 \\
&+ 75582720x^2 - 100776960x^1 + 60466176
\end{aligned}
$$



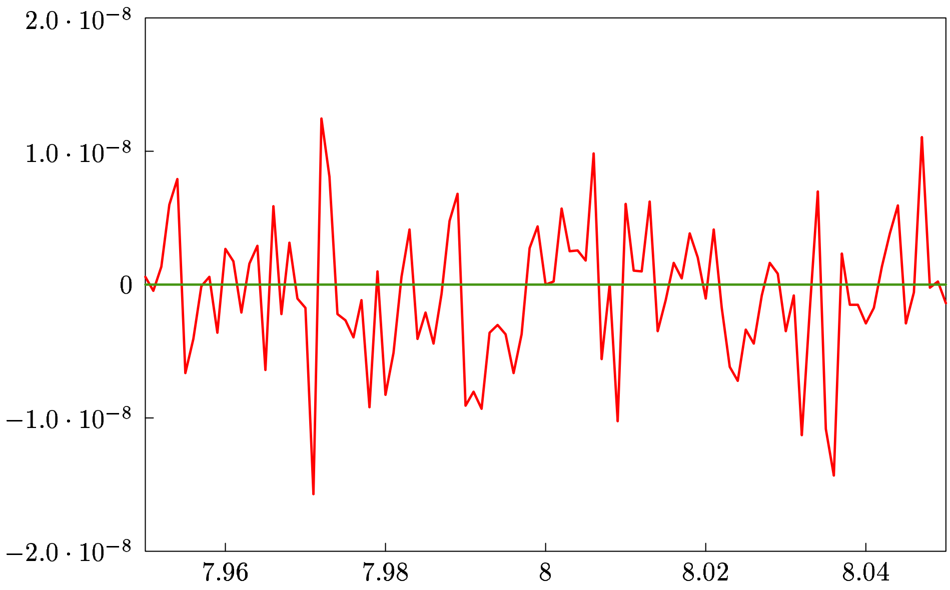


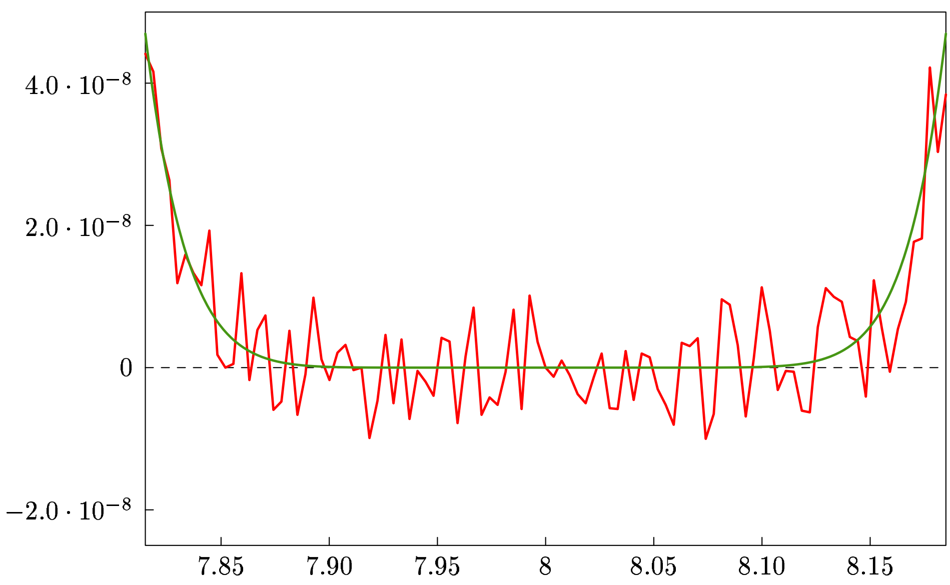
$$(x-7)^{10} = x^{10} - 70x^9 + 2205x^8 - 41160x^7 + 504210x^6 \\ - 4235364x^5 + 24706290x^4 - 98825160x^3 \\ + 259416045x^2 - 403536070x^1 + 282475249$$



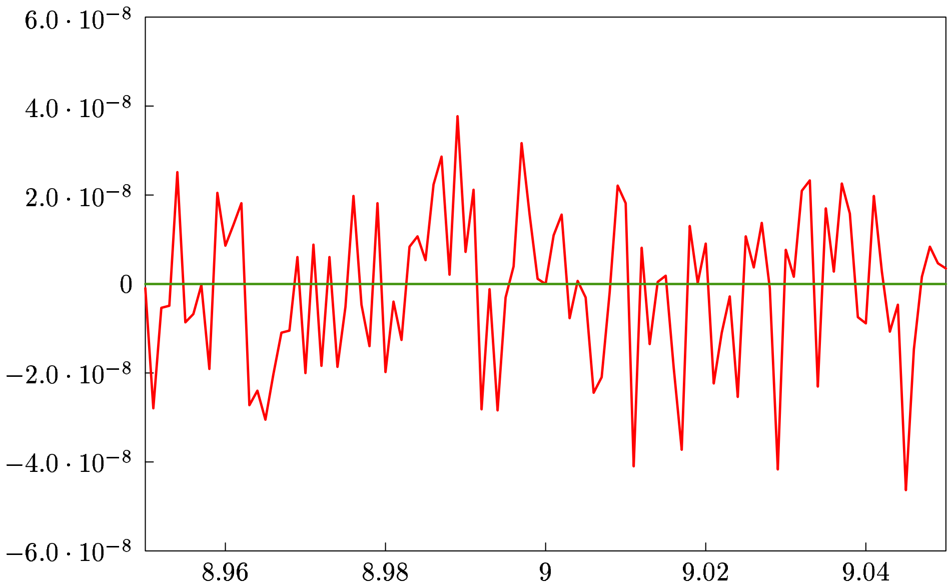


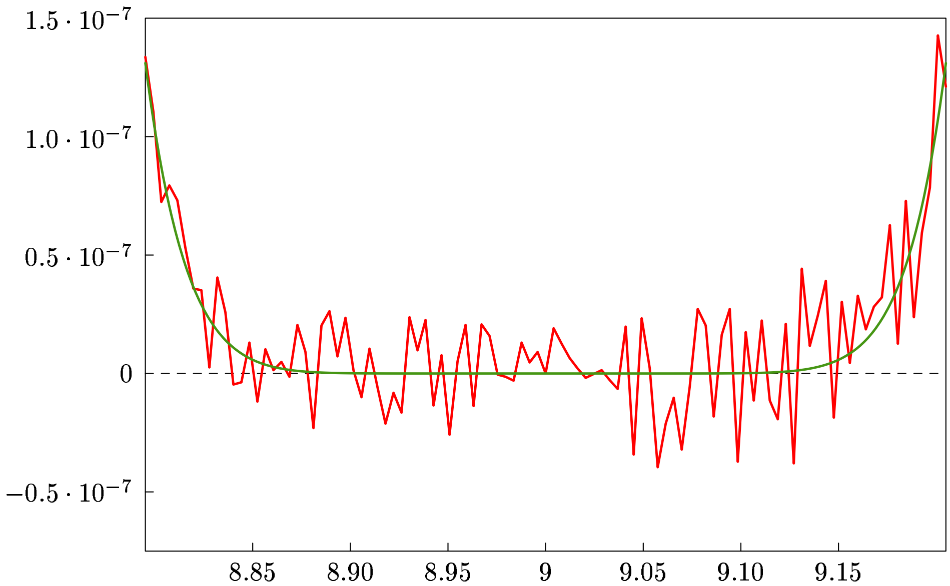
$$
\begin{aligned}
(x-8)^{10} = x^{10} &- 80x^9 + 2880x^8 - 61440x^7 + 860160x^6 \\
&- 8257536x^5 + 55050240x^4 - 251658240x^3 \\
&+ 754974720x^2 - 1342177280x^1 + 1073741824
\end{aligned}
$$



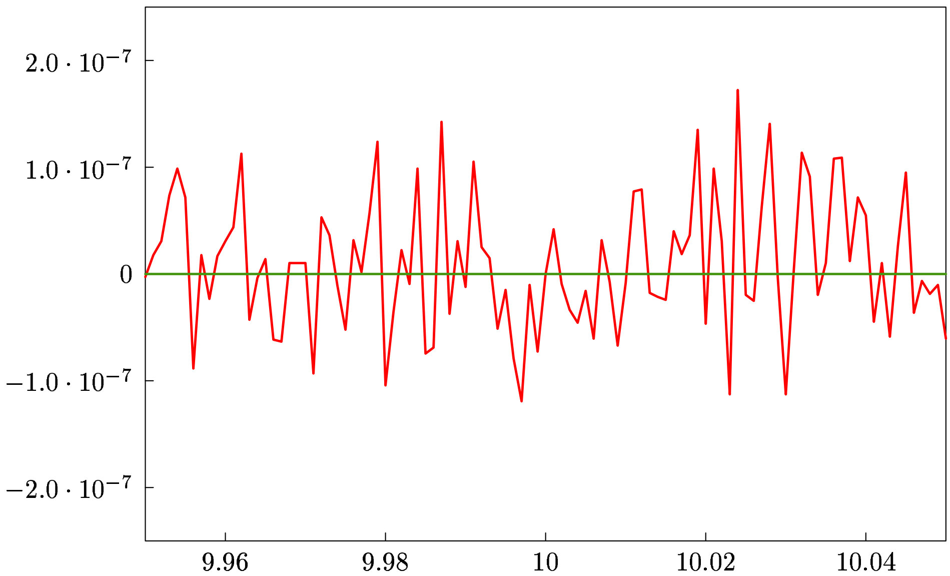


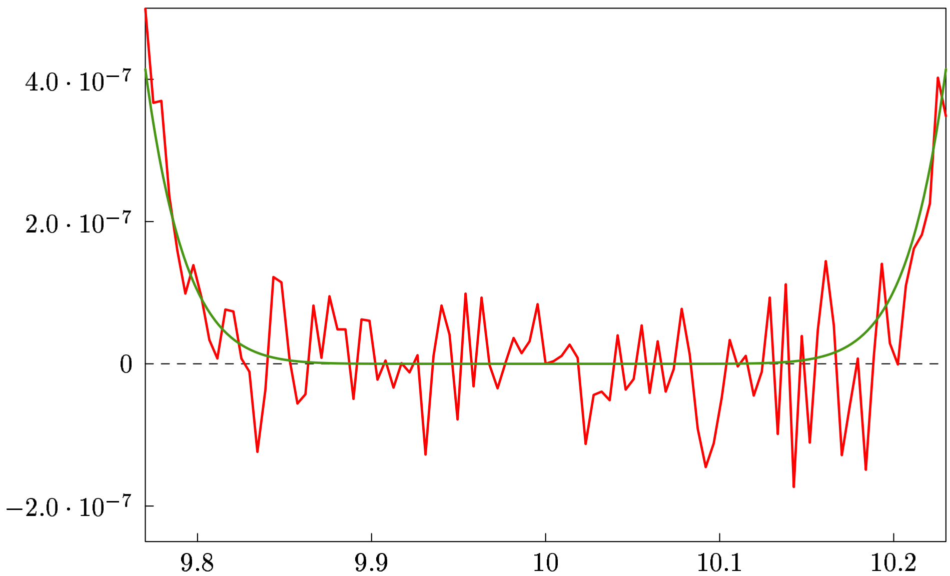
$$\begin{aligned}(x-9)^{10} = x^{10} &- 90x^9 + 3645x^8 - 87480x^7 \\ &+ 1377810x^6 - 14880348x^5 + 111602610x^4 \\ &- 573956280x^3 + 1937102445x^2 \\ &- 3874204890x^1 + 3486784401\end{aligned}$$





$$
\begin{aligned}
(x-10)^{10} = x^{10} &- 100x^9 + 4500x^8 - 120000x^7 \\
&- 25200000x^5 + 210000000x^4 + 2100000x^6 \\
&- 1200000000x^3 + 4500000000x^2 \\
&- 10000000000x^1 + 10000000000
\end{aligned}
$$





Primjer 2.

Poredak operacija nije beznačajan. Zadan je linearni sustav

$$\begin{aligned} 0.0001\, x_1 + x_2 &= 1 \\ x_1 + x_2 &= 2. \end{aligned}$$

Jedinstveno rješenje

$$x_1 = 1.\dot{0}00\dot{1}, \quad x_2 = 0.\dot{9}99\dot{8}.$$

Rješavanje računalom koje ima 4 decimalne znamenke mantise i 2 znamenke eksponenta, onda njegovo rješenje ovisi o poretku jednadžbi. Sustav zapisan u takvom računalu pamti se kao

$$\begin{aligned} 0.1000 \cdot 10^{-3}\, x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.1000 \cdot 10^1 \\ 0.1000 \cdot 10^1\, x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.2000 \cdot 10^1. \end{aligned}$$

Množenjem prve jednadžbe s 10^4 i oduzimanjem od druge, dobivamo drugu jednadžbu:

$$(0.1000 \cdot 10^1 - 0.1000 \cdot 10^5)\, x_2 = 0.2000 \cdot 10^1 - 0.1000 \cdot 10^5.$$

Računalo mora oduzimati: manji eksponent postaje jednak većem, a mantisa se denormalizira. Dobivamo

$$0.1000 \cdot 10^1 = \cdots = 0.0001 \cdot 10^4 = 0.0000|1 \cdot 10^5.$$

Zadnju znamenku nemamo gdje spremiti! Mantisa postaje 0.

Slično je i s desnom stranom. Zbog toga druga jednadžba postaje

$$-0.1000 \cdot 10^5\, x_2 = -0.1000 \cdot 10^5,$$

pa joj je rješenje

$$x_2 = 0.1000 \cdot 10^1.$$

Uvrštavanjem u prvu jednadžbu, dobivamo:

$$0.1000 \cdot 10^{-3}\, x_1 = 0.0000,$$

pa je

$$x_1 = 0,$$

što nije niti približno točan rezultat.

Promijenimo li poredak jednadžbi, dobivamo

$$\begin{aligned} 0.1000 \cdot 10^1 \quad x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.2000 \cdot 10^1 \\ 0.1000 \cdot 10^{-3}\, x_1 + 0.1000 \cdot 10^1\, x_2 &= 0.1000 \cdot 10^1. \end{aligned}$$

Množenjem prve jednadžbe s 10^{-4} i oduzimanjem od druge, dobivamo

$$(0.1000 \cdot 10^1 - 0.1000 \cdot 10^{-3})\, x_2 = 0.1000 \cdot 10^1 - 0.2000 \cdot 10^{-3},$$

pa se prethodna jednadžba svede na

$$0.1000 \cdot 10^1\, x_2 = 0.1000 \cdot 10^1.$$

Njeno rješenje je

$$x_2 = 0.1000 \cdot 10^1.$$

Uvrštavanjem u prvu jednadžbu izlazi

$$0.1000 \cdot 10^1\, x_1 = 0.1000 \cdot 10^1,$$

tj.

$$x_1 = 0.1000 \cdot 10^1,$$

što točan rezultat korektno zaokružen na četiri decimalne znamenke. $\square$

Primjer 3.

Kako se pravilno računaju neki opasni izrazi?

Treba izračunati

$$y = \sqrt{x+\delta} - \sqrt{x}, \quad x > 0,$$

a δ poznat i

$$|\delta| \ll x.$$

Deracionalizacijom:

$$\begin{aligned} y &= \sqrt{x+\delta} - \sqrt{x} \\ &= \left(\sqrt{x+\delta} - \sqrt{x}\right) \cdot \frac{\sqrt{x+\delta} + \sqrt{x}}{\sqrt{x+\delta} + \sqrt{x}} \\ &= \frac{\delta}{\sqrt{x+\delta} + \sqrt{x}}, \end{aligned}$$

Treba izračunati

$$y = \cos(x+\delta) - \cos x,$$

a x, δ poznati i

$$|\delta| \ll x.$$

Korištenjem:

$$y = -2\sin\frac{\delta}{2}\sin\left(x + \frac{\delta}{2}\right).$$

□

Primjer – Promašaj raketa Patriot.

U Zaljevskom ratu Patriot rakete nisu uspjele oboriti Scud.

Razlog katastrofe: vrijeme u računalu brojilo se desetinkama sekunde proteklim od trenutka od kad je računalo upaljeno. Desetinka u binarnom prikazu:

$$0.1_{10} = (0.0\dot{0}01\dot{1})_2.$$

Realne brojeve u tom računalu prikazivali su korištenjem nenormalizirane mantise duljine 24 bita. Spremanjem 0.1 u registar Patriot računala, napravljena je greška približno jednaka $9.5 \cdot 10^{-8}$.

Zbog stalne opasnosti od napada Scud raketama, računalo je bilo u pogonu 100 sati – ukupna greška nastala greškom zaokruživanja je

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

Scud putuje brzinom $1676\,\mathrm{m/s}$, pogreška u položaju veća od pola kilometra. □

Primjer – Eksplozija Ariane 5.

37 sekundi nakon lansiranja izvršila je samouništenje. Razlog katastrofe: program je pokušao pretvoriti preveliki 64-bitni realni broj u 16-bitni cijeli broj. Računalo je javilo grešku, što je izazvalo samouništenje. □

Primjer – Potonuće naftne platforme Sleipner A.

Naftna platforma Sleipner A potonula je prilikom sidrenja. Razlog katastrofe: kod projektiranja platforme upotrijebljena metoda konačnih elemenata s nedovoljnom točnošću. □

Zaključak

Ponašanje rezultata se može predvidjeti — strah je nepotreban, a oprez nužan. Imamo li sumnju da je oduzimanjem brojeva došlo do kraćenja, svakako treba pokušati problem preformulirati ili provjeriti drugom metodom i/ili u višoj točnosti. Jasno je da treba poznavati i greške metode.

Primjer. Taylorovi redovi za e^x i $\sin x$ oko 0 konvergiraju za proizvoljan $x \in \mathbb{R}$. Zbrajanjem dovoljno mnogo članova – u principu možemo dobro aproksimirati vrijednosti funkcija.

Ako to napravimo računalom, rezultat će biti zanimljiv.

Greška metode: Taylorov red aproksimiramo Taylorovim polinomom

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

uz grešku

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

ξ neki broj između 0 i x. Traženi Taylorovi polinomi su

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

Vrijedi

$$(e^x)^{(n)} = e^x, \qquad (\sin x)^{(n)} = \sin\Big(x + \frac{n\pi}{2}\Big),$$

pa su pripadne greške odbacivanja

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

Za $x \geq \xi > 0$ dobivamo ocjene

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

$$|R_{2n+3}(x)| = \left| \frac{\sin(\xi + \frac{2n+3}{2}\pi) x^{2n+3}}{(2n+3)!} \right| \leq \left| \frac{x^{2n+3}}{(2n+3)!} \right|.$$

Prvi odbačeni član ispod zadane točnosti $\varepsilon = 10^{-17}$ – greška odbacivanja manja ili jednaka

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

Izračunajmo

$$\sin(15\pi), \quad e^{15\pi}, \quad \sin(25\pi) \quad \text{i} \quad e^{25\pi}$$

korištenjem Taylorovog reda oko 0 u **`extended`** preciznosti.

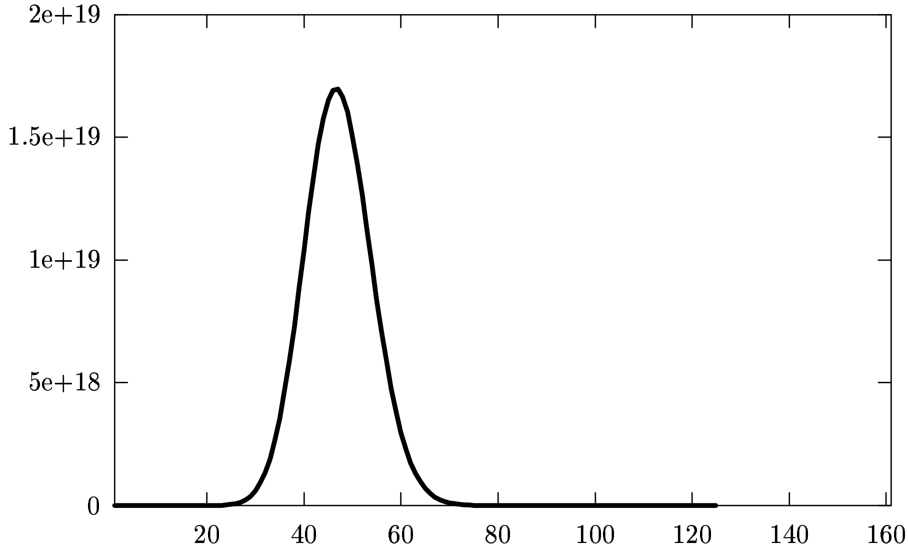
Primjer je izabran tako da je

$$\sin(15\pi) = \sin(25\pi) = 0,$$

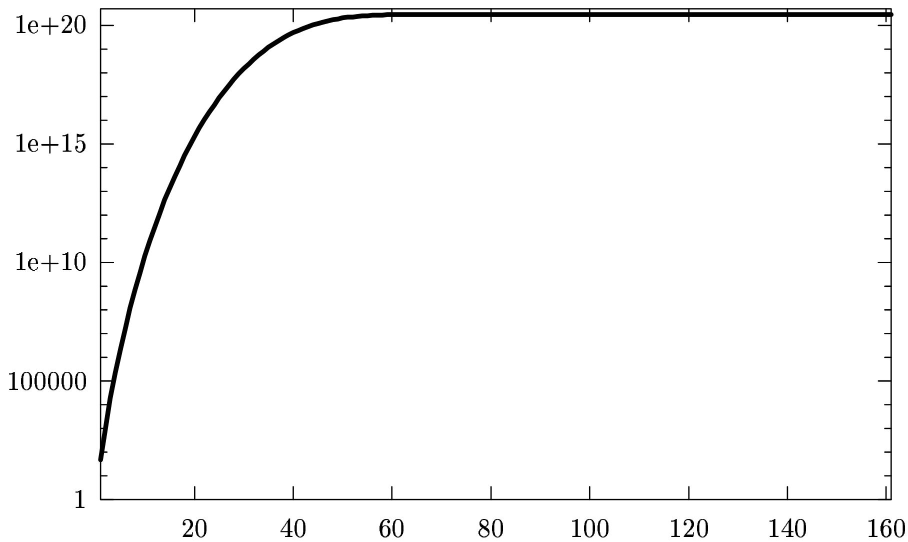
dok su druga dva broja vrlo velika.

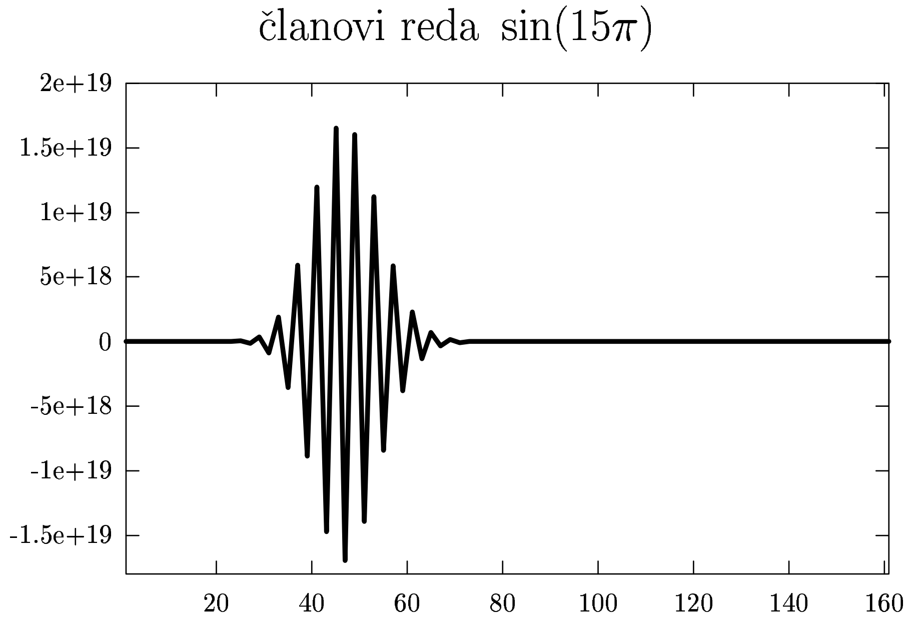
$$\begin{aligned}
\sin(15\pi)_{\mathit{funkcija}} &= 9.3241\cdot 10^{-18}\\
\sin(15\pi)_{\mathit{Taylor}} &= -2.8980\cdot 10^{-1}\\
|\mathit{greška\ odbacivanja}| &\le 2.7310\cdot 10^{-19}\\
\mathit{relativna\ greška} &= 3.1081\cdot 10^{16}\\
|\mathit{maksimalni\ član}| &= 1.6969\cdot 10^{19}\\
\\
\sin(25\pi)_{\mathit{funkcija}} &= 1.6697\cdot 10^{-17}\\
\sin(25\pi)_{\mathit{Taylor}} &= 3.0613\cdot 10^{13}\\
|\mathit{greška\ odbacivanja}| &\le 5.8309\cdot 10^{-19}\\
\mathit{relativna\ greška} &= 1.8334\cdot 10^{30}\\
|\mathit{maksimalni\ član}| &= 5.7605\cdot 10^{32}\\
\\
\exp(15\pi)_{\mathit{funkcija}} &= 2.9218\cdot 10^{20}\\
\exp(15\pi)_{\mathit{Taylor}} &= 2.9218\cdot 10^{20}\\
|\mathit{greška\ odbacivanja}| &\le 2.7600\cdot 10^{2}\\
\mathit{relativna\ greška} &= 1.4238\cdot 10^{-18}\\
|\mathit{maksimalni\ član}| &= 1.6969\cdot 10^{19}\\
\\
\exp(25\pi)_{\mathit{funkcija}} &= 1.2865\cdot 10^{34}\\
\exp(25\pi)_{\mathit{Taylor}} &= 1.2865\cdot 10^{34}\\
|\mathit{greška\ odbacivanja}| &\le 2.3782\cdot 10^{16}\\
\mathit{relativna\ greška} &= 7.0013\cdot 10^{-19}\\
|\mathit{maksimalni\ član}| &= 5.7943\cdot 10^{32}.
\end{aligned}$$

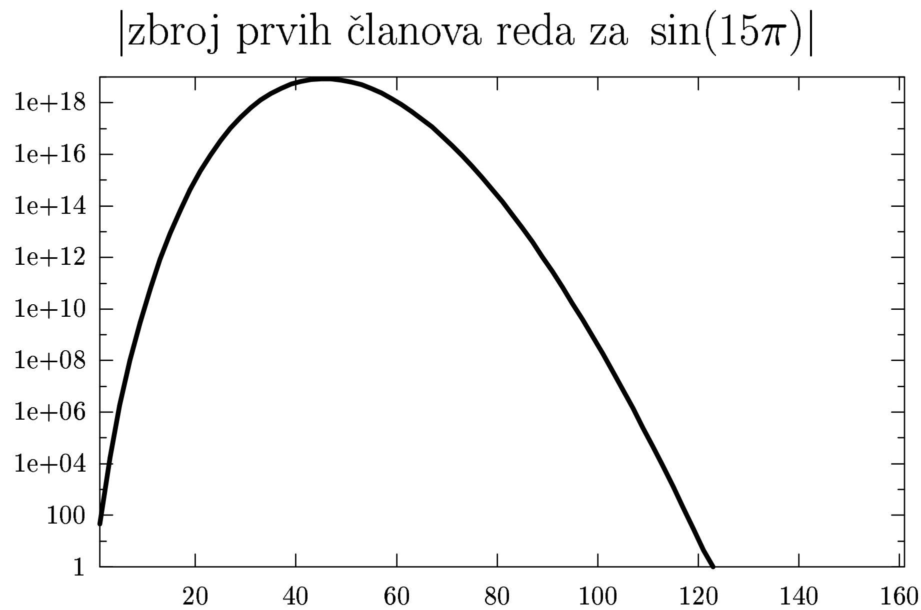
članovi reda $e^{15\pi}$



$|$zbroj prvih članova reda za $e^{15\pi}|$







Objašnjenje — očito. Za $\sin x$, rezultat je malen broj koji je dobiven – oduzimanjem velikih brojeva, pa je netočan. Kod e^x, uvijek imamo zbrajanje brojeva istog predznaka, pa je rezultat točan. $\square$