1. *Matematinis modeliavimas ir inžinerinių uždavinių sprendimas*

1.1.Fizinio reiškinio matematinis modelis ir jo sprendinys

Skaitiniai metodai yra svarbi taikomosios matematikos šaka, kurios sparčią plėtrą sąlygoja audringas kompiuterių bei jų programinės įrangos tobulėjimas.

Kompiuteriai sudarė galimybę skaitiniams metodams tapti pagrindine įvairių fizikos, mechanikos, ekonomikos, inžinerijos ir kitų sričių uždavinių sprendimo priemone, mat analiziniai matematiniai sprendimo metodai sudėtingesniems matematiniams modeliams dažnai yra nepritaikomi arba tokių metodų apskritai nėra.

Tiriant fizinius reiškinius, pirmiausiai, yra sudaromi jų matematiniai modeliai.

Matematinis modelis, tai rinkinys formulių, aprašančių esmines fizinio reiškinio ar proceso savybes matematinėmis išraiškomis. Tam tikra prasme tai galima pateikti tokia funkcine priklausomybe

$$\begin{matrix} Priklausomas \\ kintamasis \end{matrix} = f\left(\begin{matrix} nepriklausomi \\ kintamieji \end{matrix}, parametrai, \begin{matrix} poveikio \\ funkcijos \end{matrix} \right) \quad , \quad (1.1)$$

čia

- ***priklausomas kintamasis*** yra savybė, kuri atspindi sistemos elgesį ar būvį;
- ***nepriklausomi kintamieji*** paprastai yra dydžiai, tokie kaip laikas ar laiko tarpas, kuriame sistemos elgesys tiriamas;
- ***parametrai*** apibūdina sistemos savybes ar struktūrą;
- ***poveikio funkcijos*** tai išoriniai poveikiai įtakojantys sistemą.

Turėdami reiškinio matematinį modelį, jį sprendžiame (nagrinėjame) skaitiniais metodais. Po to, gautas sprendinys analizuojamas, kaip jis atitinka realią tikrovę.

Jei gautas sprendinys nėra adekvatus tikrovei, reiškinio matematinis modelis yra tikslinamas ir vėl ieškomas jo sprendinys. Kitaip tariant, reiškinio tyrimas yra iteracinis procesas.

Konkreti (1.1) matematinė išraiška gali kisti nuo paprastos algebrinės išraiškos iki didelės, sudėtingos diferencialinių lygčių sistemos. Pavyzdžiui, antrasis Niutono dėsnis sako, kad kūno pagreitis yra proporcingas jį veikiančiai jėgai. Antrojo Niutono dėsnio matematinė išraiška arba modelis yra gerai žinoma formulė

$$F=ma, \qquad (1.2)$$

čia F- kūną veikianti jėga (N arba kg·m/s^2), m-kūno masė (kg), a-kūno pagreitis (m/s^2).

Šio dėsnio (1.1) forma būtų

$$a=F/m, \qquad (1.3)$$

kurioje

a- priklausomas kintamasis apibūdinantis sistemos elgesį,

F- poveikio funkcija,

m- parametras nusakantis sistemos savybes.

(1.3) lygtyje nepriklausomų kintamųjų nėra, nes nenagrinėjame kaip pagreitis kinta laike. Ši lygtis turi keletą bendrų charakteristikų būdingų matematiniams modeliams, aprašantiems fizinius reiškinius:

- gamtos reiškinį aprašo matematine išraiška,
- pateikia idealizuotą tikrovę, t.y. modelis įvertina tik esminius gamtos reiškinius. Pavyzdžiui, antrasis Niutono dėsnis neįvertina reliatyvumo, kuris nėra svarbus nagrinėjant kūnus ir išorinius poveikius žemės paviršiuje ar arti jo, esant mažiems greičiams.
- atkuria reiškinio rezultatus ir gali būti naudojama pirminei analize. Pavyzdžiui, jei žinoma kūną veikianti jėga ir kūno masė, naudojant (1.3), galima apskaičiuoti kūno pagreitį.

Kadangi (1.2) lygties algebrinė išraiška yra paprasta, tai lengva rasti jos sprendinį. Tačiau kitų fizinių reiškinių matematiniai modeliai gali būti žymiai sudėtingesni ir dažnai tiksliai (analitiškai) neišsprendžiami.

Sudėtingesnio modelio pavyzdys būtų: rasti arti žemės paviršiaus laisvai krintančio kūno greitį, taikant antrąjį Niutono dėsnį. Tarkim, kad tai yra besileidžiantis parašiutininkas.

Šiame modelyje pagreitis išreiškiamas greičio pokyčiu per laiko vienetą (dv/dt), todėl, remdamiesi (1.3) išraiška, gausime

$$\frac{dv}{dt}=\frac{F}{m} \qquad (1.4)$$

čia v yra greitis (m/s). Jei atstojamoji jėga F, veikianti parašiutininką, yra teigiama, tai nusileidimas greitės, jei neigiama – lėtės, o jei lygi nuliui,- greitis bus pastovus.

Toliau atstojamąją jėgą išreikšime jėgų dedamosiomis. Besileidžiančio kūno atstojamąją jėgą sudaro dvi priešingos krypties jėgos: žemyn nukreipta žemės traukos jėga F_D ir jai priešinga kryptimi veikianti oro pasipriešinimo jėga F_V:

$$F=F_D+F_V\,. \qquad (1.5)$$

Žemyn veikiančios žemės traukos jėgos kryptį laikysime teigiama, todėl

$$F_D=mg \qquad (1.6)$$

čia $g\approx 9.8\text{m/s}^2$ yra laisvai krintančio kūno pagreitis.

Oro pasipriešinimo jėga veikia priešinga kryptimi ir yra proporcinga greičiui. Vadinasi,

$$F_V=-cv \qquad (1.7)$$

čia parametras c –pasipriešinimo koeficientas (kg/s), priklausantis nuo krintančio kūno savybių, kaip antai: formos, paviršiaus ploto. Parametras c gali būti funkcija nuo parašiutininko kostiumo ar jo orientacijos leidimosi metu.

Įvertinus (1.5)-(1.7) formules, (1.4) formulę užrašome taip:

$$\frac{dv}{dt}=\frac{mg-cv}{m} \qquad (1.8)$$

arba

$$\frac{dv}{dt}=g-\frac{c}{m}v\,. \qquad (1.9)$$

(1.9) lygtis yra matematinis modelis, susiejantis krintančio kūno pagreitį su kūną veikiančiomis jėgomis. Tai diferencialinė lygtis, nes mus dominantis kintamasis išreikštas išvestine (dv/dt). Priešingai nei (1.3), tikslus (1.9) sprendinys negali būti rastas naudojant paprastus algebrinius veiksmus. Analitiniam sprendiniui rasti tenka naudoti integralinį ir diferencialinį skaičiavimą.

Laikydami, kad parašiutininkas pradiniu laiko momentu nejuda (v=0,kai t=0), tai prie šių pradinių sąlygų diferencialinės lygties sprendinys būtų toks:

$$v(t)=\frac{gm}{c}(1-e^{-(c/m)t})\,. \qquad (1.10)$$

Pažymėsim, kad (1.10) sprendinys yra (1.1) bendros lygties atskiras atvejis, kur $v(t)$- priklausomas kintamasis, t- nepriklausomas kintamasis, c ir m – parametrai, o g – poveikio funkcija.

Pavyzdys. Analitinis besileidžiančio parašiutininko greičio skaičiavimas

Uždavinio formulavimas. Parašiutininkas, kurio masė 68.1kg, iššoka iš oro baliono gondolos. Apskaičiuokime parašiutininko greitį iki parašiuto išsiskleidimo, laikydami, kad pasipriešinimo koeficientas lygus 12.5kg/s.

Sprendimas. Įstatę parametrų m, c, ir g reikšmes į (1.10), turėsime tokią parašiutininko greičio formulę

$$v(t)=\frac{9.8(68.1)}{12.5}(1-e^{-(12.5/68.1)t})=53.39(1-e^{-0.18355t})\,.$$

Pasinaudodami šia formule, apskaičiuokime parašiutininko greitį diskrečiais laiko momentais. Skaičiavimo rezultatai surašyti 1.1 lentelėje.

Iš rezultatų matyti, kad pradžioje parašiutininkas greitėja, bet po pakankamai ilgo laiko tarpo greitis tampa pastovus, nes parašiutininką veikianti atstojamoji jėga pasidaro lygi nuliui.

(1.10) lygtis vadinama analitiniu arba tiksliu sprendiniu, kadangi ji gauta tiksliai išsprendus (1.9) diferencialinę lygtį. Tačiau egzistuoja aibė matematinių modelių, kuriuos išspręsti analitiniais metodais neįmanoma. Vienintelis kelias jiems spręsti yra skaitinis sprendimas, kuris aproksimuoja tikslų sprendinį.

1.1 lentelė. Parašiutininko greitis diskrečiais laiko momentais

$t[s]$	$v[m/s]$

0	0.00
2	16.40
4	27.77
6	35.64
8	41.10
10	44.87
12	47.49
∞	53.39

Norėdami skaitiniu metodu išspręsti (1.9) lygtį, turėtume pasinaudoti vienu iš šiame vadovėlyje išnagrinėtų paprastųjų diferencialinių lygčių sprendimo metodų. Pavyzdžiui, taikydami Oilerio metodą, t.y. $\frac{dv}{dt}$ aproksimuodami santykiu

$$\frac{dv}{dt} \cong \frac{\Delta v}{\Delta t} = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i}, \qquad (1.11)$$

čia Δv ir Δt – greičio ir laiko pokyčiai, $v(t_i)$- greitis laiko momentu t_i, $v(t_{i+1})$ – greitis laiko momentu t_{i+1}, turėsime diferencialinės lygties aproksimaciją

$$\frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} = g - \frac{c}{m} v(t_i).$$

Tada parašiutininko greitis diskrečiais laiko momentais bus apskaičiuojamas pagal formules:

$$t_{i+1} = t_i + \Delta t,$$

$$v(t_{i+1}) = v(t_i) + [g - \frac{c}{m} v(t_i)](t_{i+1} - t_i) \ . \qquad (1.12)$$

Pavyzdys. Skaitinis besileidžiančio parašiutininko greičio skaičiavimas

Uždavinio formulavimas. Parašiutininkas, kurio masė 68.1kg, iššoka iš oro baliono gondolos. Apskaičiuokime parašiutininko greitį iki parašiuto išsiskleidimo, laikydami, kad pasipriešinimo koeficientas lygus 12.5kg/s ir imdami laiko žingsnį $\Delta t = t_{i+1} - t_i = 2s$.

Sprendimas. Skaičiuosime pagal (1.12) formules. Kadangi $v(0)=0$, tai, $v(2)$ reikšmę apskaičiuosime pagal formulę

$$v(2) = 0 + (9.8 - \frac{12.5}{68.1} v(0)) \cdot 2 = 19.6 m/s,$$

Kai t=4, apskaičiuosime $v(4)$:

$$v(4) = 19.60 + [9.8 - \frac{12.5}{68.1} v(2)]2 = 32.00 m/s.$$

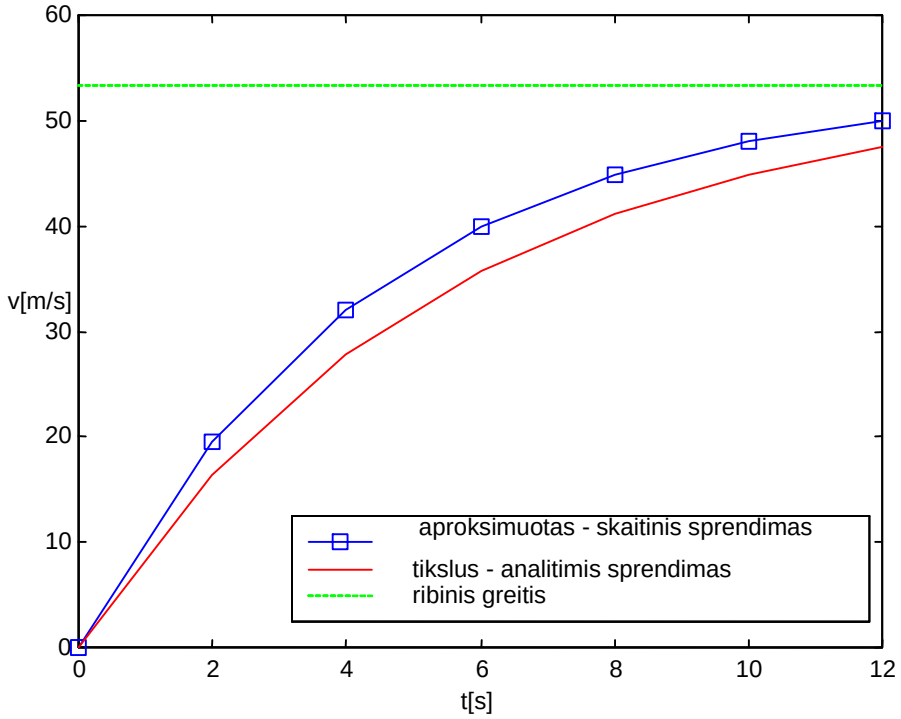
Analogiškai tęsiant skaičiavimą gauname rezultatus, kurie surašyti 1.2 lentelėje.

1.2 lentelė. Skaitiniais metodais apskaičiuotas parašiutininko greitis

t[s]	v[m/s]
0	0.00
2	19.60
4	32.00
6	39.85
8	44.82
10	47.97
12	49.96
∞	53.39

Tikslaus ir skaitinio (apytikslio) skaičiavimo rezultatai pavaizduoti (1.1) paveikslėlyje. Matome, kad skaičiavimo rezultatai, gauti naudojant skaitinius metodus, skiriasi nuo tikslaus

sprendimo rezultatų. Skirtumas paaiškinamas tuo, kad tolydinę funkciją aproksimavome tiesės atkarpomis. Norėdami šį skirtumą sumažinti, turime parinkti mažesnį laiko intervalo žingsnį.



1.1 *pav. Besileidžiančio parašiutininko greičio skaičiavimo rezultatai analitiniu ir skaitiniu metodais*

1.2 *Skaitinė analizė, metodai ir algoritmai*

Skaitinė analizė, tai matematinis (teorinis) skaitinių metodų nagrinėjimas. Pakankamai išsamią skaitinę analizę galima atlikti ir nesinaudojant kompiuteriu. Iš tikrųjų dauguma skaitinių metodų buvo sukurti žymiai anksčiau nei atsirado kompiuteriai. Matematikai, inžinieriai ir kiti mokslininkai, kuriems nepavykdavo rasti tikslaus sprendinio, buvo priversti atlikti nuoseklius veiksmus, norėdami gauti analitinio sprendinio aproksimaciją. Skaičiavimų rankomis imlumas vertė tobulinti metodus didinant jų tikslumą bei mažinat pastangas, būtinas rezultatui gauti. To pasėkoje susikūrė nauja matematikos mokslo šaka – skaitinė analizė. Skaitinė analizė ypatingą dėmesį skiria bendram skaitinių metodų 'elgesiui', bet ne jų taikymui sprendžiant atskirus uždavinius.

Skaitinė analizė padeda tyrėjui pasirinkti tinkamą metodą ir jį realizuojantį algoritmą. Ji leidžia nustatyti metodo taikymo ribas, įvertinti skaičiavimo rezultatų teisingumą bei gilesnį metodo teorinį supratimą.

Nors sąvoka "***skaitiniai metodai***" dažnai tapatinama su terminu "skaičiavimo algoritmas", tačiau iš tikrųjų taip nėra.

Skaitinis metodas, tai matematinis atliekamų skaičiavimų aprašymas.

Algoritmas – tai tiksli atliekamų veiksmų seka, būtina pasiekti norimą sprendinį.

Šis skirtumas reiškia, kad metodas gali būti realizuotas skirtingais algoritmais. Be to, galimas daiktas, kad vienas algoritmas, kuriuo tai požiūriu, yra geresnis už kitą algoritmą, realizuojantį tą patį skaitinį metodą.

Praktikoje sprendinys dažnai pasiekiamas naudojant skirtingus metodus, be to kiekvieno metodo algoritmas gali būti skirtingas. Tolesniuose skyriuose pateikti algoritmai atrinkti pagal tris subjektyvius kriterijus: supratimo lengvumą, patikimumą ir reikalaujamą kompiuterio resursų apimtį.

Sudarytą algoritmą reikia užkoduoti, t.y. parašyti šį algoritmą realizuojančią programą. Programų tekstai labai priklauso nuo pasirinktos duomenų struktūros:

programa,- tai algoritmas plius duomenų struktūra.

Pasirinkus skirtingas duomenų struktūras, gaunami skirtingi to paties algoritmo programų tekstai.

1.3. Skaičiavimo paklaidos

Su paklaidomis susidūrėme jau pirmame pavyzdyje, kur besileidžiančio parašiutininko greitis skaičiuojamas analitiniu ir skaitiniu metodais. Nors rezultatai, gauti naudojant skaitinį metodą, artimi tikslaus analitinio sprendinio rezultatams, tačiau egzistuoja paklaidos atsiradusios dėl aproksimacijos. Kadangi minėtame pavyzdyje turime ir tikslaus analitinio sprendimo rezultatus, tai skaitinio rezultato paklaidos nesunkiai gali būti įvertintos. Tačiau daugelyje praktinių inžinerinių uždavinių rasti analitinį sprendinį yra neįmanoma. Todėl nors bendru atveju tiksliai nustatyti skaitinio rezultato paklaidas yra neįmanoma, tačiau jų įvertinimo uždavinys yra labai aktualus. Egzamino ar namų darbų atveju už padarytas klaidas esame vertinami pažymiu. Darbinėje veikloje klaidos kainuoja žymiai daugiau ir gali būti katastrofiškos. Konstrukcijai ar įrengimai suirus, galimi tragiški įvykiai.

Apibrėšime skaičiaus absoliutinę ir santykinę paklaidas.

Skaičiaus absoliučioji paklaida yra tikslios ir apytikslės jo reikšmės skirtumas:

$$\Delta x = x - \bar{x}\ ;$$

čia Δx ***— absoliučioji paklaida, x — tiksli skaičiaus reikšmė,*** $\bar{x}$ ***— apytikslė skaičiaus reikšmė.***

Skaičiaus santykinė paklaida yra jo absoliučiosios paklaidos ir apytikslės reikšmės santykis:

$$\delta x = \frac{\Delta x}{\bar{x}}.$$

Pastaba. *Aišku, kad skaičiaus absoliučioji ir santykinė paklaidos turi ženklą, tačiau, norėdami apskaičiuoti didžiausią galimą reiškinio paklaidą, imame šių paklaidų modulius, t.y.*

$$\Delta x = \left|x - \bar{x}\right| \text{ ir } \delta x = \frac{\Delta x}{\left|\bar{x}\right|}.$$

Skaičiuojant dažniausiai susiduriama su trejopomis paklaidomis:

a) pradinėmis informacijos paklaidomis,
b) sprendimo metodo (aproksimavimo) paklaidomis,
c) apvalinimo paklaidomis.

Be minėtų klaidų dar galimos klaidos tiesiogiai nesusijusios su skaitiniais metodais: ***grubios klaidos ir modelio sudarymo klaidos.*** Šių klaidų čia nenagrinėsime.

1.3.1 Pradinės informacijos paklaidos

Pirmasis pradinės informacijos paklaidų šaltinis yra ***matavimų paklaidos***. Nagrinėjant realius fizinius uždavinius, jų pradiniai duomenys yra įvairių dydžių matavimo rezultatai. Kadangi jų absoliučiai tiksliai išmatuoti neįmanoma, tai jie visi turi matavimo paklaidas.

Antrasis pradinės informacijos paklaidų šaltinis yra ***iracionalieji skaičiai***, pavyzdžiui, π ,e, $\sqrt{2}$ ir kt. Šių skaičių negalima išreikšti baigtiniu ženklų skaičiumi. Pavyzdžiui, skaičius π - apskritimo ilgio ir skersmens santykis - užrašomas taip: π=3,14159265358979238462643… ir t.t. Kadangi kompiuteris skaičiavimus atlieka su baigtiniu reikšminių skaitmenų kiekiu, tai šių dydžių niekada negalima visiškai tiksliai atvaizduoti. Prarasti reikšminiai skaitmenys ir sudaro ***pradinės informacijos*** paklaida.

Skaičiuojant kompiuteriu, be šių paklaidų, atsiranda ***papildomų paklaidų***, nes kompiuteriuose vartojama dvejetainė skaičiavimo sistema. Kadangi ne visos dešimtainės trupmenos dvejetainėje sistemoje užrašomos baigtiniu pavidalu, o kompiuteryje skaičiams vaizduoti skiriamas baigtinis skaičius skilčių, tai juos vaizduojant kompiuteryje atsiranda papildoma paklaida.

Pavyzdžiui, $(0,1)_{10} = (0,000110011001100...)_2$; čia indeksas žymi skaičiavimo sistemos pagrindą. Dvejetainėje skaičiavimo sistemoje skaičius 0,1 užrašomas begaline trupmena. Dėl anksčiau paminėtos priežasties 0,1 vaizduojant kompiuteryje, atsiranda papildoma paklaida.

1.3.2 Sprendimo metodo paklaidos.

Sprendimo metodo (aproksimavimo) paklaidų šaltinis, – tai tikslių matematinių operacijų, sprendimo metodų ir dydžių aproksimavimas skaitiniuose metoduose.

Sprendimo metodo paklaidų panaikinti neįmanoma. Jas galima sumažinti tik naudojant tikslesnį metodą. Šių paklaidų analizė yra skirtinga kiekvienam skaitiniam metodui.

Panagrinėkime keletą pavyzdžių.

1. Teiloro eilutė. Teiloro eilutė plačiai naudojama skaitiniuose metoduose. Teiloro formulė įgalina bet kurią tolydinę ir be galo diferencijuojamą funkciją aproksimuoti polinomu.

Pavyzdžiui, funkcijos $y = f(x)$ pirmos eilės aproksimacija išreiškiama formule

$$f(x) \approx f(x_i) + f'(x_i)(x - x_i). \qquad (2.1)$$

Išraiškos (2.1) rezultatas yra tiesė, parodanti funkcijos didėjimą ar mažėjimą tarp x_i ir x taškų. Nors (2.1) formulė ir parodo funkcijos pokytį, tačiau ji tiksli tik tiesės atveju. Norint įvertinti funkcijos kreivumą, reikia imti pirmuosius tris Teiloro eilutės narius:

$$f(x\ ) \cong f(x_i) + f'(x_i)(x\ - x_i) + \frac{f''(x_i)}{2!}(x\ - x_i)^2\ . \qquad (2.2)$$

Paėmę *n* Teiloro eilutės narių, turėsime:

$$f(x\ ) = f(x_i) + f'(x_i)(x\ - x_i) + \frac{f''(x_i)}{2!}(x_i - x_i)^2 + \ldots + \frac{f^{(n)}(x_i)}{n!}(x\ - x_i)^n + R_n. \quad (2.3)$$

Liekamasis narys R_n įvertina visų likusių narių sumą, ir jo išraiška Lagranžo formoje yra:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x\ - x_i)^{n+1}, \qquad (2.3)$$

čia ξ - dydis tarp x_i ir x.

Duotai x reikšmei pagal (2. 3) formulę tiksliai apskaičiuoti R_n reikšmę yra neįmanoma, nes bendru atveju ξ reikšmė nėra žinoma. Tačiau tai nėra didelė blogybė, nes, parenkant algoritmą, žymiai svarbiau yra žinoti aproksimavimo paklaidos įtaką negu jos tikslią reikšmę.

1 pavyzdys. Funkcijos 1/(1-x) aproksimavimas Teiloro eilute.

Panagrinėkime funkcijos $f(x) = \frac{1}{1-x}$ aproksimacijas $P_1(x), P_2(x)$ ir $P_3(x)$, imdami šios funkcijos Teiloro eilutės, išskleistos taško x_0 =1,6 aplinkoje, atitinkamai du, tris ir keturis narius:

$$P_1(x) = \frac{1}{1-x_0} + \frac{x-x_0}{(1-x_0)^2},\quad P_2(x) = \frac{1}{1-x_0} + \frac{x-x_0}{(1-x_0)^2} + \frac{(x-x_0)^2}{(1-x_0)^3},$$

$$P_3(x) = \frac{1}{1-x_0} + \frac{x-x_0}{(1-x_0)^2} + \frac{(x-x_0)^2}{(1-x_0)^3} + \frac{(x-x_0)^3}{(1-x_0)^4}.$$

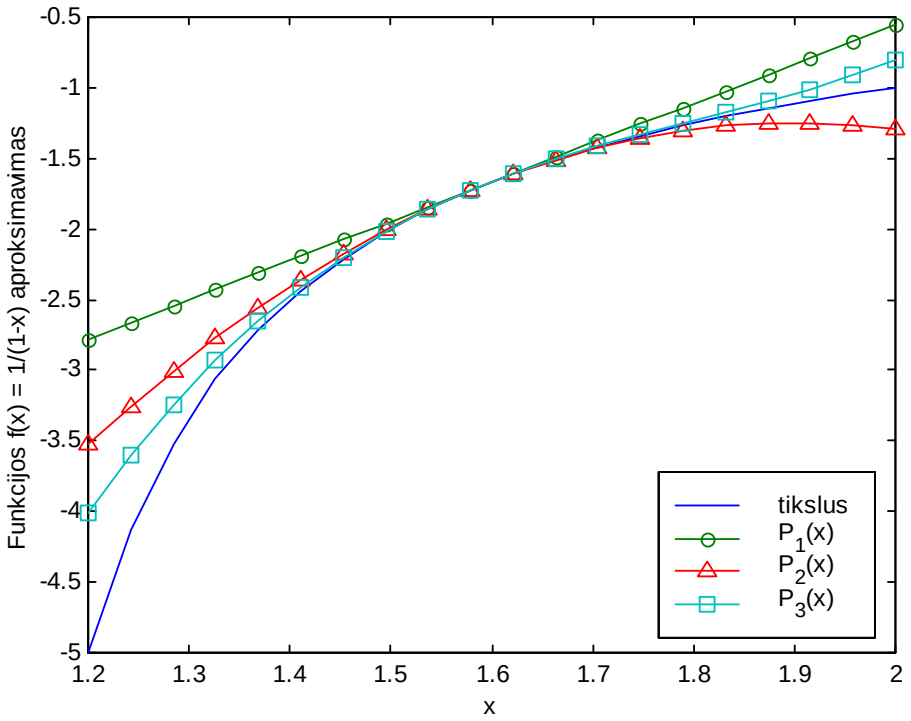
Funkcijos $f(x) = \frac{1}{1-x}$ ir jos aproksimacijų $P_1(x), P_2(x)$ ir $P_3(x)$ grafikai intervale $1.2 \le x \le 2$ pateikti 1.2 paveikslėlyje.

2 pavyzdys. Funkcijos y=sin(x) aproksimavimas Makloreno eilute.

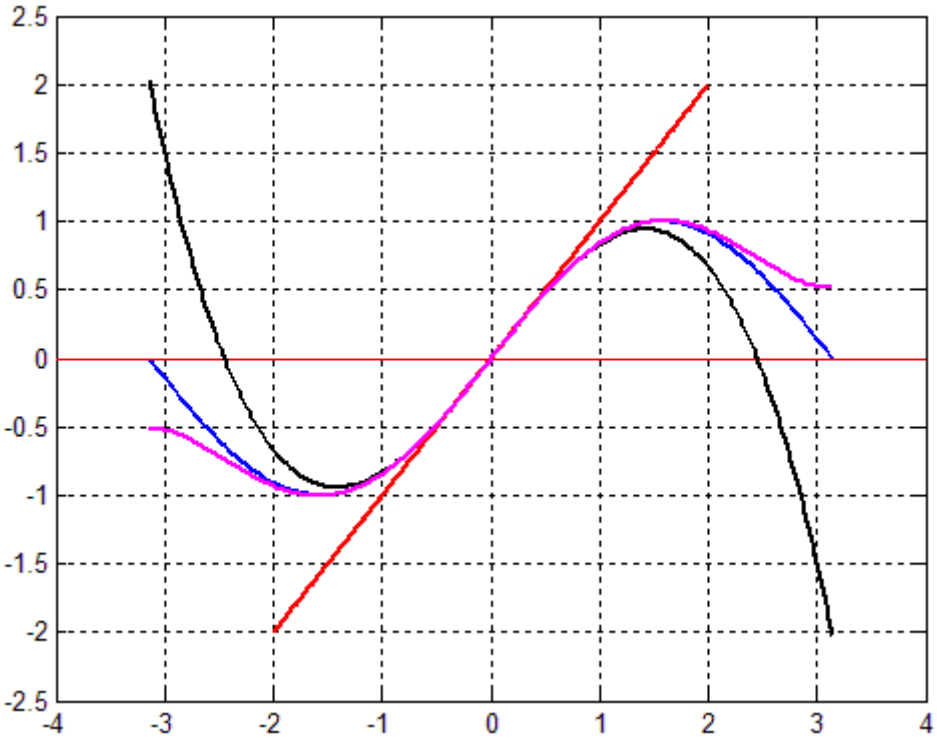
Panagrinėkime funkcijos *y=sin(x)* aproksimacijas $y_1(x), y_2(x)$ ir $y_3(x)$, imdami šios funkcijos Makloreno eilutės atitinkamai vieną, du ir tris narius:

$$y_1(x) = x,\quad y_2(x) = x - \frac{x^3}{3!},\quad y_3(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!}.$$

1.3 paveikslėlyje pavaizduoti funkcijos *y=sin(x)* ir jos aproksimacijų $y_1(x), y_2(x)$ ir $y_3(x)$ grafikai intervale $-\pi \le x \le \pi$. Funkcijos *y=sin(x)* grafikas pavaizduotas mėlyna spalva, funkcijos $y_1(x)$ grafikas pavaizduotas raudona spalva, funkcijos $y_2(x)$ grafikas pavaizduotas juoda spalva ir funkcijos $y_3(x)$ grafikas pavaizduotas rusva spalva.



1.2 pav. Funkcijos $f(x)=\frac{1}{1-x}$ *ir jos aproksimacijų* $P_1(x), P_2(x)$ *ir* $P_3(x)$ *grafikai*



1.3 pav. Funkcijos y=sin(x) ir jos aproksimacijų $y_1(x), y_2(x)$ *ir* $y_3(x)$ *grafikai*

2. ***Skaitinis integravimas.*** Apskaičiuojant apibrėžtinius integralus, yra naudojamos įvairios kvadratūrinės formulės. Šios formulės ir jų taikymo strategijos yra nagrinėjamos skaitinio integravimo skyriuje.

2 ***pavyzdys.*** *Apskaičiuokime reikšmę apibrėžtinio integralo* $R=\int_0^{\frac{\pi}{2}}\sin x dx$ *vieną kartą pritaikę ketvirtosios eilės Niutono ir Koteso formulę.*

Sprendimas. Turėsime:

$$R=\int_0^{\frac{\pi}{2}}\sin x dx \approx \frac{2}{45}h\left(7(\sin 0+\sin\frac{\pi}{2})+32(\sin\frac{\pi}{8}+\sin\frac{3\pi}{8})+12\sin\frac{\pi}{4}\right)=0.99999156547299,$$

čia $h=\frac{\pi}{8}$.

Tiksli šio integralo reikšmė yra 1. Skirtumas tarp tikslios ir apskaičiuotos integralo reikšmės yra: *pkl=1-0.99999156547299=8.434527007272763e-006.*

Ketvirtosios eilės Niutono ir Koteso kvadratūrinės formulės paklaida yra:

$$K_4=-\frac{8}{945}h^7 f^{(6)}(x), x\in[a,b].$$

Nors bendru atveju šia formule naudotis nėra patogu, nes integravimo intervale reikia įvertinti pointegralinės funkcijos šeštosios eilės išvestinės modulio didžiausią reikšmę, tačiau pateikto pavyzdžio atveju šios išvestinės modulio didžiausia reikšmė yra lygi vienetui. Tada

$$|K_4|\le\frac{8}{945}\left(\frac{\pi}{8}\right)^7 = 1.219206806072575\text{e-}005.$$

Kaip matome, realioji paklaida, kaip ir turėjo būti, yra mažesnė už didžiausią teoriškai apskaičiuotą paklaidą.

1.3.3 Apvalinimo paklaidos

Apvalinimo paklaidų šaltinis yra skaičių vaizdavimas kompiuteryje: skaičiai kompiuteryje rašomi dvejetainėje skaičiavimo sistemoje ir atvaizduojami baigtiniu dvejetainių skilčių (ženklų) skaičiumi.

Skaičių vaizdavimas kompiuteryje. Dėl techninių priežasčių kompiuteriuose skaičiams vaizduoti yra skirtas baigtinis skilčių skaičius. Kitaip tariant, skaičiai kompiuteriuose vaizduojami baigtiniu tikslumu. Tai padidina skaičiavimo greitį ir sumažina apimtį atminties, reikalingos jiems vaizduoti. Tačiau toks skaičių vaizdavimas iššaukia ir labai nepageidaujamą efektą – ***apvalinimo paklaidas***, kurios mažina rezultatų tikslumą. Tai susiję su jaunesnių skilčių praradimu atliekant skaičiavimo operacijas. Norint suprasti apvalinimo paklaidų prigimtį, būtina žinoti skaičių saugojimo kompiuteryje teorinius pagrindus.

Dvejetainė skaičiavimo sistema. Beveik visi kompiuteriai naudoja dvejetainę skaičiavimo sistemą, t.y. skaičiams vaizduoti naudojami ne dešimt simbolių kaip įprastoje dešimtainėje skaičiavimo sistemoje, bet du simboliai: 0 ir 1. Pavyzdžiui, dešimtainėje sistemoje skaičius 27 reiškia 2*10+7. Tuo tarpu šis skaičius dvejetainėje skaičiavimo sistemoje būtų vaizduojamas skaičiumi 11011, kuris reikštų $1*2^4+1*2^3+0*2^2+1*2^1+1*2^0$.

Dešimtainių skaičių pervedimas į dvejetainę skaičiavimo sistemą.

Norint dešimtainį skaičių užrašyti dvejetainėje skaičiavimo sistemoje, reikia šį skaičių išreikšti dvejeto laipsnių suma.

Sveikąjį skaičių dvejeto laipsnių suma išreikšime šį skaičių bei gautus dalmenis nuosekliai dalydami iš 2 iki paskutinysis dalmuo bus lygus vienetui. Dalybos eigoje įsidėmėkime gautas liekanas. Tada, užrašę paskutinį dalmenį ir gautas liekanas atvirkščia tvarka, turėsime dešimtainį skaičių dvejetainėje skaičiavimo sistemoje.

Pavyzdžiui, dešimtainį skaičių 27 dvejetainėje skaičiavimo sistemoje apskaičiuosime taip:

1) 27 dalome iš 2; turėsime dalmenį 13 ir liekaną 1,

2) 13 dalome iš 2; turėsime dalmenį 6 ir liekaną 1,

3) 6 dalome iš 2; turėsime dalmenį 3 ir liekaną 0,
4) 3 dalome iš 2; turėsime dalmenį 1 ir liekaną 1.

Užrašę paskutinį dalmenį ir gautas liekanas atvirkščia tvarka turėsime: 11011.
Tai ir yra dešimtainis skaičius 27 dvejetainėje skaičiavimo sistemoje, t.y. $(27)_{10}=(11011)_2$, čia indeksas parodo skaičiavimo sistemos pagrindą.

Norėdami dešimtainę trupmeną pervesti į dvejetainę skaičiavimo sistemą (išreikšti šią trupmeną dvejeto neigiamų laipsnių suma), šią trupmeną dauginame iš dviejų. Toliau, gautos sandaugos trupmeninę dalį vėl dauginame iš 2 ir taip toliau. Tada gautų sandaugų sveikosios dalys ir bus po kablelio einančių dvejetainės trupmenos skilčių reikšmės. Aišku, kad sandaugos procesas bendru atveju gali būti begalinis. O tas reiškia, kad ne visas dešimtaines trupmenas galime paversti baigtinėmis dvejetainėmis trupmenomis.

1 pavyzdys. Dešimtainę trupmeną 0.5 galime užrašyti baigtine dvejetaine trupmena.
Sprendimas. Padauginę 0.5 iš 2, turėsime 1.0.
Kadangi sandaugos trupmeninė dalis lygi nuliui, tai, toliau ją dauginant iš 2, sandauga visada bus lygi nuliui.
Vadinasi, $(0.5)_{10}=(0.1)_2$.

2 pavyzdys. Dešimtainę trupmeną 0.1 perveskime į dvejetainę skaičiavimo sistemą.
Sprendimas. Skaičių 0.1 padauginę iš 2, turėsime 0.2. Šios sandaugos sveikoji dalis yra lygi nuliui, o trupmeninė dalis 0.2.
Padauginę šią trupmeninę dalį iš 2, turėsime 0.4. Taigi, sandaugos sveikoji dalis vėl yra lygi nuliui, o trupmeninė dalis 0.4.
Toliau, padauginę trupmeninę dalį iš 2, turėsime 0.8. Šios sandaugos sveikoji dalis yra lygi nuliui, o trupmeninė dalis 0.8.
Dar kartą trupmeninę dalį padauginę iš 2, turėsime 1.6. Šios sandaugos sveikoji dalis yra lygi vienetui, o trupmeninė dalis – 0.6.
Tęsdami sandaugų trupmeninių dalių daugybą iš 2, turėsime tokias sandaugas: 1.2; 0.4; 0.8; 1.6; 1.2; 0.4 ir t.t.
Vadinasi, $(0.1)_{10}=(0.0001100110\ldots)_2$.
Kadangi sandaugų trupmeninė dalis niekada nebus lygi nuliui, tai reiškia, kad skaičiaus $(0.1)_{10}$ dvejetainėje sistemoje yra begalinis.
1.3 lentelėje parodyti dešimtainių skaičių pavyzdžiai ir jų pervedimas į dvejetainę skaičiavimo sistemą

1.3 Lentelė. Dešimtainių skaičių pavyzdžiai ir jų pervedimas į dvejetainę skaičiavimo sistemą.

Pagrindas 10	*Pervedimas*		*Pagrindas 2*
1	1 =	2^0	0000 0001
2	2 =	2^1	0000 0010
4	4 =	2^2	0000 0100
8	8 =	2^3	0000 1000
9	8 + 1 =	$2^3 + 2^0$	0000 1001
10	8 + 2 =	$2^3 + 2^1$	0000 1010
27	16 + 8 + 2 + 1 =	$2^4 + 2^3 + 2^1 + 2^0$	0001 1011
			vienas baitas

Pastaba. Skaičius iš dešimtainės į dvejetainę skaičiavimo sistemą galima pervesti žymiai greičiau, jei kaip tarpinė sistema naudojama aštuntainė skaičiavimo sistema. Dešimtainis skaičius, pirmiausia pervedamas į 8-ainę sistemą, o po to, automatiškai, perkoduojamas į dvejetainę sistemą. Skaičius iš 8-ainės sistemos į dvejetainę perkoduojamas kiekvieną aštuntainį skaitmenį pakeičiant jo

dvejetainiu ekvivalentu: $(0)_8=(000)_2$, $(1)_8=(001)_2$, $(2)_8=(010)_2$, $(3)_8=(011)_2$, $(4)_8=(100)_2$, $(5)_8=(101)_2$, $(6)_8=(110)_2$, $(7)_8=(111)_2$. Pavyzdžiui, $(37)_{10}=(45)_8=(100101)_2$.

3 pavyzdys. Pasinaudodami tarpine 8-aine skaičiavimo sistema, skaičių $(57,25)_{10}$ perveskime iš dešimtainės skaičiavimo sistemos į dvejetainę skaičiavimo sistemą.

Sprendimas. Kadangi sveikieji ir trupmeniniai skaičiai iš vienos sistemos į kitą pervedami skirtingais būdais, tai skaičių $(57,25)_{10}$ skaidome į sveikąją ir trupmeninę dalis.

Aišku, kad $(57)_{10}=(71)_8=(111001)_2$, o $(0,25)_{10}=(0.2)_8=(0,010)_2$.

Vadinasi, $(57,25)_{10}=(111001,010)_2$.

Bitas, baitas ir žodis. Kompiuteriuose skaičiams charakterizuoti naudojamos sąvokos: ***bitas, baitas ir žodis.*** Bitas, baitas ir žodis,- tai dvejetainiai skaičiai, turintys skirtingą skilčių skaičių.

Bitas, tai dvejetainis skaičius turintis vieną skiltį, t.y. 0 arba 1.

Baitas, tai aštuonių bitų grupė, t.y. dvejetainis skaičius turintis 8 skiltis.

Žodis, tai mažiausias adresuojamas kiekvieno kompiuterio atminties vienetas. Žodžio ilgis priklauso nuo kompiuterio architektūros ir elementinės bazės, o taip pat nuo operacinės sistemos. Pavyzdžiui, kompiuterių specialistai kalba apie "64-bitų mikroprocesorių" ir "32-bitų operacinę sistemą". Pirmoji sąvoka charakterizuoja kompiuterio architektūrą ir elementinę bazę, o antroji – operacinę sistemą.

Sveikųjų skaičių vaizdavimas. Sveikieji skaičiai kompiuteryje vaizduojami tiksliai. Aritmetiniai veiksmai tarp sveikųjų skaičių taip pat tikslūs su išlyga, kad rezultatas neviršija didžiausio leistino sveikojo skaičiaus (dalmuo yra sveikasis skaičius, t.y. liekana atmetama).

Tarkime menamo kompiuterio žodžio ilgis yra 4 bitai. Pagrindiniai terminai aprašantys žodį yra: kairysis bitas dešinysis bitas

3 2 1 0 ←*bito numeris*

0 1 0 1 ←***sveikasis skaičius (žodis)***

3 2 1 0 ←*bito numeris*

vyriausias bitas *jauniausias bitas*

Pateiktasis dvejetainis skaičius yra

$0*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 5.$

Naudojant šį 4 bitų žodį galima saugoti 16 sveikųjų (pagrindu 10) skaičių: nuo $(0)_{10}$ iki $(15)_{10}$ arba $(0000)_2$ iki $(1111)_2$, čia indeksas parodo skaičiavimo sistemos pagrindą.

Atimties operacija kompiuteryje keičiama neigiamo skaičiaus sumavimu. Neigiamo skaičiaus požymiui žodyje skiriamas kairysis (vyriausias) bitas:

0xxx – teigiamas sveikasis skaičius,

1xxx – neigiamas sveikasis skaičius.

Todėl diapazonas sveikųjų skaičių, kuriuos galima atvaizduoti 4 bitais yra nuo -8=1000 iki 7=0111 arba

←perpildymas perpildymas →

-8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7

I I

$-2^{(4-1)}$ $2^{(4-1)}-1$

Beveik visuose kompiuteriuose sveikiesiems skaičiams saugoti skiriama 16 bitų (2 baitai) arba 32 bitai (4 baitai). Didžiausias sveikasis skaičius, kurį galima saugoti kaip16 bitų dvejetainį skaičių, yra $2^{16}-1=65535$.

Didžiausia MATLAB'o sveikojo skaičiaus (didžiausias sveikasis skaičius žymimas *bitmax*) reikšmė patalpinta 2.2 lentelėje *(Specialūs MATLAB'o kintamieji)* ir yra lygi: $bitmax=2^{53}-1$.

Realiųjų skaičių vaizdavimas. Realusis skaičius x kompiuteriuose vaizduojamas slankiojo kablelio išraiška:

$x = \pm mp^{\pm q}$; čia p — skaičiavimo sistemos pagrindas,

m — mantisė, tenkinanti sąlygą $\frac{1}{p} \le m < 1$,

q — skaičiaus eilė (sveikasis skaičius).

Tokią skaičiaus x išraišką žymėsime $fl(x)$.

Dvejetainėje sistemoje slankaus kablelio žodis skirstomas į segmentus: 1 bitas skiriamas vaizduoti skaičiaus x ženklą, r bitų skiriama skaičiaus x eilei, o t bitų skiriama skaičiaus x mantisės trupmeninei daliai. Tada galime laikyti, kad eilės q viršutinis rėžis $u = p^r - 1$, o apatinis — $l = -\left(p^r - 1\right)$.

Viengubo tikslumo (*single precision*) slankaus kablelio skaičiai saugomi 32 bitų žodyje, o ***dvigubo tikslumo*** (*double precision*) 64 bitų žodyje.

Tikslumą apsprendžia kiek skilčių skiriama slankaus kablelio skaičiui ir kaip skiltys paskirstytos tarp mantisės ir eilės. Šis paskirstymas priklauso nuo kompiuterio tipo, tačiau dažniausiai ***viengubo tikslumo*** slankaus kablelio skaičiaus mantisei yra skiriami 24 bitai, o eilei – 8 bitai, o ***dvigubo tikslumo*** slankaus kablelio skaičiaus mantisei yra skiriami 53 bitai, o 11 bitų yra skiriami eilei.

Išnagrinėkime, kiek skirtingų realiųjų skaičių kompiuteris gali įsiminti; kitaip tariant, kokia yra kompiuterio realiųjų skaičių aibės galia G.

Kompiuterio realiųjų skaičių aibės galią apskaičiuojama pagal formulę:

$$G = 2 \cdot \left(p - 1\right) \cdot p^{t-1} \cdot \left(u - l + 1\right) + 1.$$

Šioje formulėje reiškinys $\left(p - 1\right) \cdot p^{t-1}$ žymi skaičių mantisių, kurias galima parašyti p skaičiavimo sistemoje, kai mantisei vaizduoti skirta t skilčių ir mantisė m tenkina sąlygą $\frac{1}{p} \le m < 1$. Daugiklis $\left(u - l + 1\right)$ rodo, kiek skirtingų reikšmių gali įgyti eilė q. Daugiklis 2 reiškia, kad skaičiai gali būti tiek teigiamieji, tiek neigiamieji. Dėmuo 1 žymi nulį. Pavyzdžiui, jeigu $p = 2$, $t = 4$, $l = -3$ ir $u = 3$, tai

$$G = 2 \cdot 1 \cdot 2^3 \cdot \left(3 + 3 + 1\right) + 1 = 113.$$

Nors kompiuteriams realiųjų skaičių aibės galia G yra labai didelis skaičius, tačiau jis yra baigtinis.

Vadinasi, begalinę kontinuumo galios realiųjų skaičių aibę kompiuteris atvaizduoja į baigtinę diskrečiųjų realiųjų skaičių aibę, kurios galia G.

Kokios yra tokio atvaizdavimo pasekmės?

1. ***Operacijų neekvivalentumas***. Aritmetiniai veiksmai ir kompiuterio atliekami tie patys matematiniai veiksmai nėra ekvivalentūs. Pavyzdžiui, matematinėje formulėje $y = a \cdot \left(b + c\right)$ sudėties ir daugybos veiksmai atliekami absoliučiai tiksliai. Tačiau šioje formulėje, užrašytoje kokia nors algoritmine kalba, pavyzdžiui, $y := a*\left(b + c\right)$, sudėtis ir daugyba bus atliekamos apytiksliai ir rezultato tikslumas priklausys nuo skaičių vaizdavimo kompiuteryje.

2. ***Aritmetinių dėsnių negaliojimas***. Kompiuterių aritmetikoje ne visada galioja asociatyvumo ir distributyvumo dėsniai.

Tarkime, kad turime kompiuterį, kuriam $p = 10$, o $t = 4$. Išnagrinėkime tapatybes:

$$a + \left(b + c\right) = \left(a + b\right) + c, \qquad (2.4)$$

$$a \cdot \left(b \cdot c\right) = \left(a \cdot b\right) \cdot c, \qquad (2.5)$$

$$a \cdot \left(b - c\right) = a \cdot b - a \cdot c. \qquad (2.6)$$

Sakykime, (2.4) tapatybės $a = 0{,}7520 \cdot 10^4$, $b = 0{,}4976 \cdot 10^0$, $c = 0{,}2897 \cdot 10^0$. Tada $a + \left(b + c\right) = 0{,}7521 \cdot 10^4$, $\left(a + b\right) + c = a = 0{,}7520 \cdot 10^4$, o tikslus rezultatas yra $0{,}75207873 \cdot 10^4$.

Tarkime, kad (2.5) tapatybės $a = 0{,}9302 \cdot 10^4$, $b = 0{,}6741 \cdot 10^0$, $c = 0{,}8544 \cdot 10^1$. Tada $a \cdot \left(b \cdot c\right) = 0{,}5358 \cdot 10^5$, $\left(a \cdot b\right) \cdot c = 0{,}5357 \cdot 10^5$, o reikšmė $a \cdot b \cdot c$ devynių ženklų tikslumu yra $0{,}535749657 \cdot 10^5$.

Tarkime, kad (2.6) tapatybės $a = 0{,}9964$, $b = 0{,}6392$, $c = 0{,}6375$. Tada $a \cdot (b - c) = 0{,}1592 \cdot 10^{-2}$, $a \cdot b - a \cdot c = 0{,}1500 \cdot 10^{-2}$, o tiksli reikšmė lygi $0{,}159188 \cdot 10^{-2}$.

3. ***Mažiausiojo ir didžiausiojo skaičiaus egzistavimas***. Kai eilei vaizduoti yra skirta *r* skilčių, tai nelygų nuliui skaičių, kurio modulis yra mažiausias, galime užrašyti taip: $\omega = p^{-1} \cdot p^{-(p^r - 1)} = p^{-p^r}$. Jei skaičiuodami gauname rezultatą, kurio modulis mažesnis už ω, tai tas rezultatas traktuojamas kaip nulis ir vadinamas ***mašininiu nuliu*** (***underflow***).

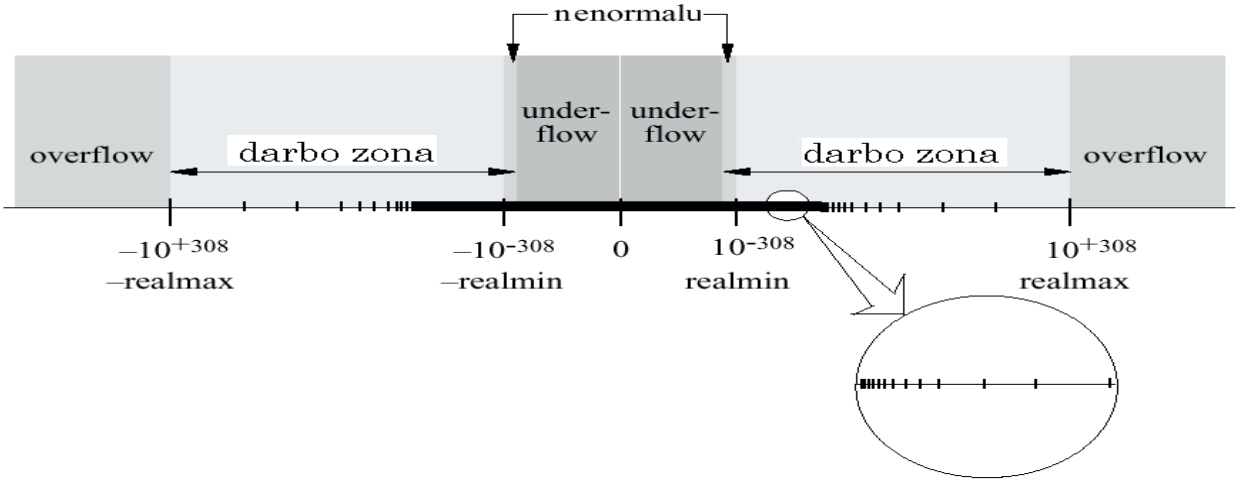
Didžiausias kompiuterio įsimenamas skaičius yra

$\Omega = \left(1 - p^{-t}\right) \cdot p^{p^r - 1}$.

Kai skaičiavimo rezultatas didesnis už šį skaičių, kompiuteris sustoja. Toks avarinis sustojimas vadinamas „***persipildymu***“ (***overflow***).

Didžiausiojo ir mažiausiojo MATLAB‘o realiųjų skaičių reikšmės patalpintos 2.2 lentelėje (*Specialūs MATLAB’o kintamieji)* ir atitinkamai yra lygūs: $\Omega = 1.797693134862316 \cdot 10^{308}$, $\omega = 2.225073858507201 \cdot 10^{-308}$.

1.4 paveiksle skaičių ašyje pavaizduota kompiuterio realiųjų skaičių aibė.



1.4 pav. Kompiuterio realiųjų skaičių aibė.

4. ***Kompiuterinis epsilon***. Labai svarbus kompiuterių aritmetikos parametras yra skaičius, kuris rodo, kiek mažiausiai du realieji skaičiai turi skirtis vienas nuo kito, kad kompiuteris juos suprastų kaip atskirus skaičius. Šis parametras yra standartizuotas ir apibrėžiamas taip: tai pats mažiausias teigiamasis skaičius, tenkinantis sąlygą $1 + eps > 1$. MATLAB‘e kompiuterinis *epsilon* žymimas ***eps*** ir yra lygus: ***eps***=2.220446049250313 $\cdot 10^{-16}$ (žr. 2.2 lentelę).

Galime teigti, kad kiekvieno realiojo skaičiaus $x \neq 0$ $fl(x)$ santykinė paklaida yra mažesnė nei *eps* ir, atvirkščiai, bet kurio realiojo skaičiaus $x \neq 0$ $fl(x) \in [x \cdot (1 - eps);\ x \cdot (1 + eps)]$.

Dviejų realiųjų skaičių $fl(x)$ ir $fl(y)$ kairiosios skilčių pusės sutampa, jei $\frac{|x - y|}{|x|} \leq \sqrt{eps}$.

Apvalinimo paklaidos priklauso nuo mantisei vaizduoti skirtų skilčių skaičiaus ir nuo skaičiavimo veiksmų. Kiekvieno aritmetinio veiksmo rezultatą galime užrašyti tokia išraiška:

$y = m_y \cdot 10^q + g_y \cdot 10^{q-t}$;

čia *t* — mantisei vaizduoti skirtų skilčių skaičius, $\frac{1}{p} \leq m_y < 1$, o $0 \leq g_y < 1$.

Pavyzdžiui, skaičių $0{,}1624 \cdot 10^3$ ir $0{,}1769 \cdot 10^1$ suma, kai $t = 4$, gali būti užrašyta taip: $0{,}1641 \cdot 10^3 + 0{,}6900 \cdot 10^{-1}$.

Gautas rezultatas apvalinamas pagal įprastas apvalinimo taisykles. Taigi

$$\bar{y} = \begin{cases} m_y \cdot 10^q, & \text{kai} \quad 0 \le g_y < 0{,}5, \\ m_y \cdot 10^q + 10^{q-t}, & \text{kai} \quad 0{,}5 \le g_y < 1. \end{cases}$$

Vadinasi, didžiausios absoliučiosios paklaidos modulis lygus $0{,}5 \cdot 10^{q-t}$, o galimos didžiausios santykinės paklaidos modulis yra

$$\frac{0{,}5 \cdot 10^{q-t}}{0{,}1 \cdot 10^q} = 5 \cdot 10^{-t}.$$

Kyla klausimas: kaip kaupiasi skaičiavimo paklaidos? Lengvai galima įrodyti, kad:

1) absoliučioji sumos paklaida lygi dėmenų absoliučiųjų paklaidų sumai plius sudėties veiksmo absoliučioji apvalinimo paklaida: $\Delta(x+y) = \Delta x + \ + \Delta y + r$; čia r — sudėties veiksmo absoliučioji apvalinimo paklaida;

2) absoliučioji skirtumo paklaida lygi turinio ir atėminio absoliučiųjų paklaidų skirtumui plius atimties veiksmo absoliučioji apvalinimo paklaida r: $\Delta(x-y) = \Delta x - \Delta y + r$;

3) absoliučioji sandaugos paklaida apskaičiuojama pagal formulę $\Delta(x \cdot y) = \bar{y} \cdot \Delta x + \bar{x} \cdot \Delta y + r$; čia r — daugybos veiksmo absoliučioji apvalinimo paklaida;

4) dalmens apvalinimo paklaida apskaičiuojama pagal formulę

$$\Delta\left(\frac{x}{y}\right) = \frac{\bar{y}\Delta x - \bar{x}\Delta y}{\bar{y}^2} + r;$$

čia r — dalybos veiksmo absoliučioji apvalinimo paklaida.

Kadangi didžiausios galimos santykinės paklaidos didumas priklauso tik nuo mantisei vaizduoti skirtų skilčių skaičiaus, tai patogiau analizuoti santykines paklaidas.

Aptarkime, kaip kaupiasi santykinės apvalinimo paklaidos, atliekant aritmetinius veiksmus. Remiantis absoliučiųjų paklaidų formulėmis, gaunamos tokios aritmetinių veiksmų santykinės paklaidos formulės:

sudėties: $\delta(x+y) = \frac{\bar{x}}{\bar{x}+\bar{y}}\delta x + \frac{\bar{y}}{\bar{x}+\bar{y}}\delta y + r,$

atimties: $\delta(x-y) = \frac{\bar{x}}{\bar{x}-\bar{y}}\delta x - \frac{\bar{y}}{\bar{x}-\bar{y}}\delta y + r,$

daugybos: $\delta(x \cdot y) = \delta x + \delta y + r,$

dalybos: $\delta\left(\frac{x}{y}\right) = \delta x - \delta y + r;$

čia r — atitinkamo aritmetinio veiksmo santykinė apvalinimo paklaida.

4 pavyzdys. Tarkim kompiuterio žodyje mantisei skirti 4 bitai. Turime du dvejetainius skaičius: x ir y. Reikia rasti jų sumą.

Sprendimas. Tarkime, kad $x = (0.1011)_2 * 2^3$ ir $y = (0.1100)_2 * 2^{-1}$.
Suvienodinus x ir y eiles turėsime: $x + y = (0.1011)_2 * 2^3 \ + \ (0.0000\underline{1100})_2 * 2^3$,
čia pabrauktos skiltys paprasčiausiai ignoruojamos, nes mantisei skirti tik 4 bitai. Vadinasi, gauta suma yra lygi: $x + y = (0.1011)_2 * 2^3$.
Išvada. Dydis y yra per mažas, kad atsispindėtų sumos rezultate.

5 pavyzdys. Didelė apvalinimo paklaida paprastoje formulėje **(**pavyzdys paimtas iš literatūros: N.J.Higham,Accuracy and Stability of Numerical Algorithms, 1996,SIAM,section 1.11)**.**

Natūrinio logaritmo pagrindą – skaičiaus e reikšmę galima apskaičiuoti pagal formulę:

$$e = e^1 = \lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n .$$

Žemiau pateikta skaičiaus e apskaičiavimo MATLAB‘o procedūra *epprox*.

```
function epprox
```

```
e =exp(1);
fprintf('\n n f(n)error \n ');
for n=logspace(0,16,9)
f =(1+1/n)^n;
fprintf('%9.1e %14.10f %14.10f \n ',n,f,abs(f-e));
end
```

Funkcijos skaičiavimo rezultatai:

n	*f*(*n*)	*paklaida*
1.0e+00	2.0000000000	0.7182818285
1.0e+02	2.7048138294	0.0134679990
1.0e+04	2.7181459268	0.0001359016
1.0e+06	2.7182804691	0.0000013594
1.0e+08	2.7182817983	0.0000000301
1.0e+10	2.7182820532	0.0000002248
1.0e+12	2.7185234960	0.0002416676
1.0e+14	2.7161100341	0.0021717944
1.0e+16	1.0000000000	1.7182818285

Kintamojo *n* reikšmei didėjant iki 10^8 skaičiaus *e* paklaida mažėja. Toliau didinant *n* reikšmę, skaičiaus *e* paklaida pradeda didėti. Kai *n* įgauna 10^{16} reikšmę, skaičiaus *e* reikšmė tampa lygi vienetui ir yra visiškai klaidinga.

Vadinasi, skaičiuojant skaičiaus *e* reikšmę, sutinkame dviejų rūšių apvalinimo paklaidas: viena mažesnę, o kitą - katastrofišką.

Mažesnės paklaidos šaltinis yra apytikslis skaičiaus 1/*n* atvaizdavimas dvejetainėje skaičiavimo sistemoje ir paklaida, susikaupusi atliekant aritmetinius veiksmus.

Katastrofiškos paklaidos šaltinis yra *kompiuterinis epsilon*. Kai $n = 10^{16}$, tai $\frac{1}{n} < eps$. Todėl $\left(1 + \frac{1}{n}\right) = 1$ ir skaičiavimo rezultatas lygus vienetui. Aišku, kad visoms *n* reikšmėms, didesnėms už 10^{16}, apskaičiuota *e* reikšmė bus lygi vienetui.

Reikia pažymėti, kad pavyzdyje nagrinėjamos paklaidos gautos atlikus tik tris matematines operacijas. Todėl apvalinimo paklaidų, sukauptų atlikus milijonus sudėties ar daugybos veiksmų, įtaka rezultatams gali būti esminė.