KAUNO TECHNOLOGIJOS UNIVERSITETAS
FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS
TAIKOMOSIOS MATEMATIKOS KATEDRA

Eglė Sokolovaitė

DIFERENCIALINIŲ LYGČIŲ KRAŠTINIO UŽDAVINIO SKIRTUMINIŲ SCHEMŲ TYRIMAS

Magistro darbas

Vadovas

doc. dr. N. Listopadskis

KAUNAS, 2012

KAUNO TECHNOLOGIJOS UNIVERSITETAS
FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS
TAIKOMOSIOS MATEMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. dr. N. Listopadskis
2012 06 02

DIFERENCIALINIŲ LYGČIŲ KRAŠTINIO UŽDAVINIO SKIRTUMINIŲ SCHEMŲ TYRIMAS

Taikomosios matematikos magistro baigiamasis darbas

Vadovas
doc. dr. N. Listopadskis
2012 06 01

Recenzentas
prof. habil. dr. R. Barauskas
2012 06 01

Atliko
FMMM 0 gr. stud.
E. Sokolovaitė
2012 05 30

KAUNAS, 2012

KVALIFIKACINĖ KOMISIJA

Pirmininkas:	Rimantas Rudzkis, profesorius (VU MII)
Sekretorius:	Eimutis Valakevičius, docentas (KTU)
Nariai:	Jonas Valantinas, profesorius (KTU)
	Vytautas Janilionis, docentas (KTU)
	Vidmantas Povilas Pekarskas, profesorius (KTU)
	Rimantas Rudzkis, habil. dr., vyriausiasis analitikas (DnB NORD Bankas)
	Zenonas Navickas, profesorius (KTU)
	Arūnas Barauskas, dr., vice-prezidentas projektams (UAB „Baltic Amadeus“)

Sokolovaitė E. Analysis of Finite Difference Schemes for Boundary Value Problems: Master‘s Work in Applied Mathematics / Supervisor doc. dr. N. Listopadskis; Department of Applied Mathematics, Faculty of Fundamental Sciences, Kaunas University of Technology. – Kaunas, 2012. –82p.

SUMMARY

Differential equations often arise in real problems, however, are usually solved only using numerical methods. Boundary value problem is even more difficult than initial value problem, as in boundary problems it is neccessary to solve system of equations. In educational literature usually only several simpler difference schemes are analysed, however, in full spectrum – from theory of stability to efficiency. But we know that it is possible to implement finite difference schemes using higher order approximations. In this paper we implement them seeking not efficiency but generality – we analyse broad accuracy spectrum of finite difference schemes.

Folowing theory, higher order approximations would lead to more accurate solutions, however, in real simulations, computer’s errors put limits to this. Besides, full theoretical analysis is also not possible when analyzing higher order schemes. In this paper there are analysed several boundary value problems: ordinary linear differential equations, partial differential equations of two types: eliptic and parabolic, with two variables.

In this paper there are implemented some general methods for solving these problems, where one can choose order of approximation. For analyzing stability of implemented schemes simple numerical analysis is chosen. For inicializing parabolic problem (initial-boundary problem) extrapolation idea is applied. For stability analysis some known properties of the solution are used to draw converging domains. Also, solution with higher order residual terms is found using only extrapolation, and comparative analysis is provided.

ĮVADAS

Diferencialinėmis lygtimis aprašoma daugelio svarbių gamtos, technikos, visuomenės reiškinių matematiniai modeliai. Tačiau šių uždavinių sprendiniai dažniausiai gali būti rasti tik skaitiniais metodais. Kraštinis diferencialinių lygčių uždavinys sprendžiamas sudėtingiau nei pradinis uždavinys, nes čia sprendinys apribojamas iš kraštų, todėl jo ieškant skaitiniu būdu, tenka spręsti lygčių sistemas. Mokomojoje literatūroje nagrinėjamos mažų tikslumo eilių schemos, jos kiek galima giliau ištiriamos teoriškai ir tuomet, turint konkretų uždavinį, realizuojamos. Tačiau aproksimuoti išvestines galima ir aukštesnių eilių skirtuminiais santykiais. Moksliniuose straipsniuose taip pat siūlomos konkrečios schemos, dažnai žemesnių tikslumo eilių, tačiau efektyvios. Šio darbo tikslas yra panagrinėti įvairių tikslumų skirtumines schemas, nesiekiant efektyvumo, bet siekiant bendrumo – ištirti kuo platesnį schemų tikslumų spektrą.

Aukštesnių tikslumų skirtuminės schemos, jei pilnai paklustų teorijai, leistų dar tiksliau apskaičiuoti sprendinius, arba skaičiuojant didesniu žingsniu (taigi ir sprendžiant lygčių sistemą iš mažiau lygčių) gauti panašius rezultatus, kaip ir naudojant mažesnio tiksumo schemas. Kompiuterinės skaičiavimų paklaidos ir saugojamos skaičių mantisės baigtinumas apriboja teorijos teikiamas galimybes. O pilnas teorinis tyrimas sudėtingoms schemoms jau yra negalimas. Darbe nagrinėjami kelių skirtingų tipų kraštiniai diferencialiniai uždaviniai – paprastųjų tiesinių diferencialinų lygčių su vienu- erdvės- kintamuoju, nestacionarus dalinių išvestinių tiesinių diferencialinių lygčių su dviem – erdvės ir laiko - kintamaisiais bei stacionarus dalinių išvestinių kraštinis uždavinys su dviem kintamaisiais.

Sukurti metodai bendru atveju apskaičiuoti išvestinių aproksimavimo skirtuminiais santykiais koeficentams, atliekama paprasta išvestinės viename taške aproksimavimo analizė, tuomet pereinama prie vienmačio uždavinio, jam spręsti ir tirti sukurti metodai, kuriuose galima pasirinkti įvairius parametrus, įskaitant norimą aproksimavimo eilę. Stabilumui tirti skaitiniu būdu pasirinktas paprastas metodas, kurį galima naudoti ir sprendžiant realų uždavinį. Tiriamam dvimačiam stacionariam uždaviniui taip pat sukurta bendra programa, kurioje galima pasirinkti norimą aproksimavimo eilę. Vienmačiam nestacionariam uždaviniui – (pradiniam – kraštiniam uždaviniui) spręsti pasiūlyta pasirenkamo tikslumo schema, kurios inicializavimui pritaikyta ekstrapoliavimo idėja, o konvergavimo sritys surastos pasinaudojant uždavinio specifika. Taip pat sukurtas metodas sprendiniui rasti aukštesniu tikslumu, naudojantis vien ekstrapoliavimo idėja; pateikta lyginamoji analizė.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

1. TEORINĖ DALIS

1.1. Kraštiniai diferencialinių lygčių uždaviniai

Kraštiniu uždaviniu diferencialinių lygčių teorijoje vadinama diferencialinė lygtis ir su ja susieti papildomi apribojimai kintamųjų kitimo sričių kraštuose, vadinami kraštinėmis sąlygomis. Tokio uždavinio sprendinys – tai duotos diferencialinės lygites sprendinys, tenkinantis tas sąlygas.

Kraštinis uždavinys turi būti suformuluotas korektiškai, t.y. taip, kad egzistuotų vienintelis sprendinys, tolydžiai priklausantis nuo kraštinių sąlygų. Tolydumo reikalavimas svarbus fizikiniuose modeliuose, nes esant fizikinių eksperimenų matavimų nedidelėms paklaidoms, nepageidaujami esminiai sprendinio pasikeitimai.

Pradinio uždavinio formuluotėje papildomos sąlygos sprendiniui užrašomos tai pačiai nepriklausomo kintamojo reikšmei – nagrinėjamos kintamojo apibrėžimo srities pradžiai. Kraštiniame uždavinyje papildomos sąlygos užrašomos nagrinėjamos kintamojo apibrėžimo srities kraštuose.

1.1.1. Diferencialinių lygčių klasifikacija

Diferencilinės lygtys skirstomos į paprastąsias ir dalinių išvestinių. Nagrinėsime tik tiesinių diferencialinių lygčių uždavinius.

Tiesinės diferencialinės lygtys su dalinėmis išvestinėmis

$$a_1 u_{xx} + a_2 u_{xy} + a_3 u_{yy} + a_4 u_x + a_5 u_y + a_6 u = f \qquad (1.1)$$

klasifikuojamos pagal $a_2^2 - 4a_1a_3$ reikšmę srityje Ω:

1. $a_2^2 - 4a_1a_3 > 0 \ \ \forall (x,y) \in \Omega$ – hiperbolinė
2. $a_2^2 - 4a_1a_3 = 0 \ \ \forall (x,y) \in \Omega$ – parabolinė
3. $a_2^2 - 4a_1a_3 < 0 \ \forall (x,y) \in \Omega$ - elipsinė

Hiperbolinės lygties pavyzdys - bangos lygtis:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \qquad (1.2)$$

Parabolinės lygties pavyzdys – šilumos laidumo lygtis:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \qquad (1.3)$$

Elipsinės lygties pavyzdys – Puasono lygtis

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y) \qquad (1.4)$$

Uždaviniai su parabolinėmis ir hiperbolinėmis lygtimis priskiriami pradiniam-kraštiniam uždaviniui; kadangi yra laiko kintamasis, tai turima pradinė sąlyga – sprendinio reikšmė pradiniu laiko momentu, ir sprendinio ieškoma su kraštiniais apribojimais iki norimo laiko momento. Uždaviniai su elipsinėmis lygtimis priskiriami kraštiniam uždaviniui. Kraštiniai uždaviniai toliau klasifikuojami pagal kraštinių sąlygų tipus.

1.1.2. Kraštinių uždavinių tipai

Dažniausiai išskiriamos trys lokalių kraštinių sąlygų tipai:

1. pirmojo tipo – Dirichle (Dirichlet) – kraštinė sąlyga. Srities krašte reikšmė suteikiama sprendiniui.
2. antrojo tipo – Neimano (Neumann)- kraštinė sąlyga. Srities krašte reikšmė suteikiama sprendinio išvestinei.
3. trečiojo tipo – Robin - kraštinė sąlyga. Srities krašte apibrėžiama sprendinio ir jo išvestinės reikšmių tiesinė kombinacija. Kitaip sakant, tai pirmo ir antro tipo kraštinių sąlygų tiesinis darinys.

Pavyzdžiui, nagrinėkime konkrečią PDL (PDL - paprastoji diferencialinė lygtis), aprašančią strypo temperatūros pasiskirstymą. Tegu strypas užima sritį $[0; l]$ ir žinoma jo paviršiaus (arba aplinkos) temperatūra T_0. Tiriamas temperatūros pasiskirstymas strypo viduje. Žymėkime $u(x)$ - temperatūra taške x. Ši funkcija tenkina diferencialinę lygtį

$$-\frac{d}{dx}\left(k(x)\frac{du}{dx}\right) + q(x)u(x) = f(x)\,,\ 0 < x < l \qquad (1.5)$$

Pirmasis narys nusako difuziją, k(x) - šilumos laidumo koeficientas, antras narys nusako šilumos mainus su aplinka, q(x) - paviršiaus šilumos mainų koeficientas, f(x) - išorinių šilumos šaltinių tankis. Priklausomai nuo to, kokie procesai vyksta strypo galuose, gaunamos įvairios kraštinės sąlygos. Pavyzdžiui,

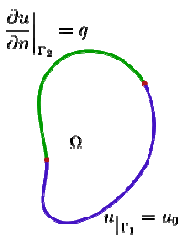
1. Jei vienas strypo galas $x = 0$ laikomas absoliutaus nulio temperatūroje, tai šiame krašte sprendiniui suteikiama žinoma reikšmė *0*, o krašte $x = l$ temperatūra μ_l , tuomet gauname pirmojo tipo kraštines sąlygas: $u(0) = 0, u(l) = \mu_l$.
2. Jei metalinio strypo gale $x = l$ yra šildytuvas, šildantis strypą pastovia temperatūra μ_l, tai pati energija būtų nežinoma, tačiau jos kitimas tame gale yra žinomas, todėl kraštinė sąlyga susiejama su ieškomo sprendinio išvestine: $k(l)u'(l) = \mu_l$ -tai antrojo tipo kraštinės sąlygos pavyzdys.
3. Jei strypo pradžioje $x = 0$ vyksta laisvi šilumos mainai su aplinka, tai temperatūra strypo pradžioje aprašoma pirmojo ir antrojo tipo kraštinių sąlygų tiesiniu dariniu: $-k(0)u'(0) + q(0)(u(0) -$

$T_0) = 0$ (lygybė nuliui, nes nėra išorinių šilumą suteikiančių šaltinių, pvz šildytuvo). Tai trečiojo tipo kraštinės sąlygos pavyzdys.

Tokių tipų sąlygos apibrėžiamos tiek PDL, tiek dalinių išvestinių diferencialinėms lygtims.

Daugiamačiu atveju – pavyzdžiui, diferencialinei lygčiai $\nabla^2 y + y = 0$, kur ∇^2 - Laplaso operatorius $\nabla^2 y = \sum_{i=1}^{n} \frac{\partial^2 y}{\partial x_i^2}$, $y = y(x_1, x_2, \dots, x_n)$ – n-matė funkcija apibrėžta srityje $\Omega \subset \mathbb{R}^n$ (n>1 atveju gaunama dalinių išvestinių diferencialinė lygtis) – kraštinių sąlygų pavyzdžiai: pirmojo tipo kraštinės sąlygos $y(x) = f(x)$, $\forall x \in \partial\Omega$, f(x) - žinoma funkcija, apibrėžta krašte $\partial\Omega$. Antro tipo kraštinės sąlygos f(x), $\forall x \in \partial\Omega$, čia $\frac{\partial y}{\partial n} = \nabla y \cdot n$ – gradiento ir krašto $\partial\Omega$ normalės skaliarinė sandauga, $x = (x_1, x_2, \dots, x_n)$. Trečiojo tipo kraštinės sąlygos krašte $\partial\Omega$ - tiesinis pirmojo ir antrojo tipų kraštinių sąlygų darinys: $ay(x) + b\frac{\partial y}{\partial n} = f, \forall x \in \partial\Omega$ bendru atveju a, b, f – x funkcijos. Vienmačiu atveju srities kraštas $\partial\Omega = \{0, l\}$ - du taškai, dvimačiu atveju – kreivė, trimačiu – paviršius.

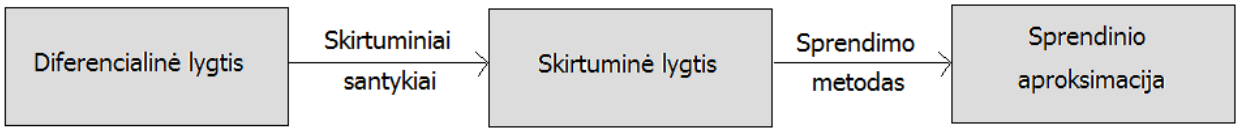
Dalinių išvestinių lygtims dar gali būti apibrėžiamos mišrios kraštinės sąlygos, kai vienoje dalyje krašto (lanke) sprendiniui ar jo dalinei išvestinei priskiriama viena reikšmė, kitoje krašto dalyje – kita reikšmė.



1.1 pav. Mišrių kraštinių sąlygų pavyzdys

Čia išvardintos kraštinės sąlygos vadinamos lokaliosiomis. Be šių išvardintų kraštinių sąlygų tipų, sutinkamos ir sudėtingesnės - nelokaliosios kraštinės sąlygos, t.y. tokios, kurios apibrėžia sąryšį tarp sprendinio ar jo išvestinės reikšmių srities kraštinuose taškuose ir srities vidiniuose taškuose, t.y. į kraštinių sąygų išraiškas įeina ir ieškomojo sprendinio reikšmės srities viduje. Gaunama lygčių sistema, kurią sudaro diferencialinė lygtis ir papildomos kraštinių sąlygų lygtys.

1.2. Baigtinių skirtumų metodas



1.2 pav. Baigtinių skirtumų metodą aprašanti schema

1.2.1. Kraštinio uždavinio suvedimas į skirtuminę schemą

Baigtinių skirtumų metodo tikslas yra supaprastinti diferencialinį uždavinį išvestines aproksimuojant baigtiniais skirtumais, tokiu būdu sprendinio radimą suvedant į algebrinių lygčių sistemos sprendimą.

Pirmiausia nagrinėjamoje srityje apibrėžiamas diskretusis tinklas. Tuomet diferencialinė lygtis aproksimuojama baigtinių skirtumų lygtimi kiekviename gauto tinklo mazge. Antrojo ir trečiojo tipo kraštinės sąlygose esančios išvestinės taip pat aproksimuojamos baigtiniais skirtumais. Aišku, ieškomos funkcijos vertė kiekviename mazge priklausys nuo greta esančių mazgų verčių, todėl gaunama lygčių sistema. Pirmiausia išsiaiškinsime, kokiais baigtiniais skirtumais gali būti aproksimuojamos išvestinės.

1.2.2. Dešinieji, kairieji ir centriniai skirtumai.

Dešininis (angl. forward) skirtumas apibrėžiamas lygybe:

$$\Delta_h f(x) = f(x+h) - f(x) \tag{1.6}$$

Kairinis (angl. backward) skirtumas:

$$\nabla_h f(x) = f(x) - f(x-h) \tag{1.7}$$

Centrinis (angl. central) skirtumas:

$$\delta_h f(x) = \frac{\Delta_h f(x) + \nabla_h f(x)}{2} \tag{1.8}$$

Nagrinėkime funkciją f(x) – kraštinio uždavinio, apibrėžto srityje Ω, sprendinį. Kai diskrečiojo tinklo žingsnis pastovus ir lygus h, tai f(x) išvestinių aproksimacijos skirtuminiais santykiais kiekviename taške gaunamos naudojantis f(x) Teiloro eilute:

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \frac{f^{(3)}(x_0)}{3!}(x-x_0)^3 + \cdots \tag{1.9}$$

Priklausomai nuo to, kuriuose mazguose imamos funkcijos reikšmės išvestinei aproksimuoti, gaunami kairiniai, centiniai ar dešininiai skirtuminiai santykiai. Pavydžiui, jei i-tojo mazgo reikšmė aproksimuojama i-tojo, i+1-ojo ir t.t. mazgų reikšmėmis, gaunami dešininiai skirtuminiai santykiai i-tojo mazgo išvestinės aproksimavimui. Funkcijos skleidinys Teiloro eilute i+1-ajame mazge $x_{i+1} = x_i + h$:

$$f(x_i + h) = f(x_i) + f'(x_i)h + \frac{f''(x_i)}{2!}h^2 + \cdots + \frac{f^{(n)}(x_i)}{n!}h^n + \frac{f^{(n+1)}(\xi)}{(n+1)!}h^{n+1} \tag{1.10}$$

Čia $x_i < \xi < x_i + h$.

Paskutinis narys $R_n(x_i + h) = \frac{f^{(n+1)}(\xi)}{(n+1)!}h^{n+1}$ žymi liekaną, kai $f(x_i + h)$ aproksimuojama baigtine suma $\sum_{k=0}^{n} \frac{f^{(k)}(x_i)}{k!}h^k$.

Paėmę aproksimaciją iki n=1, t.y.

$$f(x_i + h) = f(x_i) + f'(x_i)h + \frac{f^{(2)}(\xi)}{2!}h^2 \qquad (1.11)$$

gauname pirmos eilės išvestinės mazge x_i aproksimaciją antruoju skirtuminiu santykiu (naudojamos 2-ų mazgų reikšmės – i-tojo ir vieno papildomo – i+1-ojo):

$$f'(x_i) = \frac{f(x_i+h)-f(x_i)}{h} - \frac{f^{(2)}(\xi)}{2!}h\,, \qquad (1.12)$$

t.y. $f'(x_i) = \frac{f(x_i+h)-f(x_i)}{h} + O(h)$, jei $f^{(2)}(x)$ aprėžta.

Panašiai aproksimuojama ir kairiniu antruoju skirtuminiu santykiu:

$$f(x_i - h) = f(x_i) - f'(x_i)h + \frac{f^{(2)}(\xi)}{2!}h^2 \Rightarrow f'(x_i) = \frac{f(x_i)-f(x_i-h)}{h} + \frac{f^{(2)}(\xi)}{2!}h. \qquad (1.13)$$

Centriniai skirtuminiai santykiai, gauti iš kairinių ir dešininių antrųjų skirtuminių santykių, duoda jau aukštesnės eilės paklaidą atžvilgiu h, ir be to, juose išvestinės mazge x_i aproksimacijai naudojami taškai $x_{i-1} = x_i - h$ ir $x_{i+1} = x_i + h$, t.y. gaunama aproksimacija trečiuoju skirtuminiu santykiu:

$$f(x_i + h) = f(x_i) + f'(x_i)h + \frac{f''(x_i)}{2!}h^2 + \frac{f^{(3)}(x_i)}{3!}h^3 + \cdots \ldots \qquad (1.14)$$

$$f(x_i - h) = f(x_i) - f'(x_i)h + \frac{f''(x_i)}{2!}h^2 - \frac{f^{(3)}(x_i)}{3!}h^3 + \cdots \qquad (1.15)$$

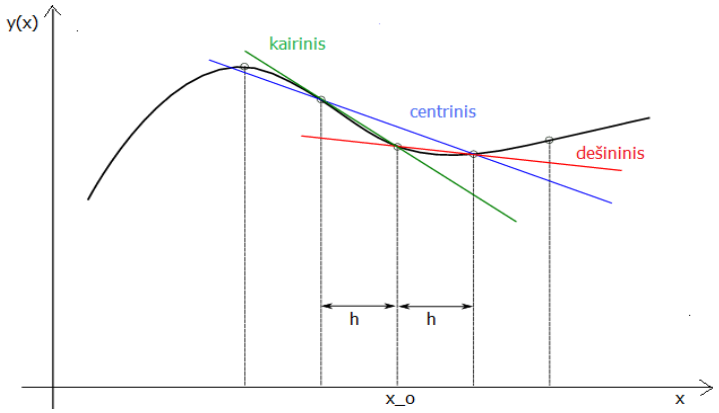
Atėmę ir išreiškę $f'(x_i)$ gauname:

$f(x_i + h) - f(x_i - h) = 2f'^{(x_i)}h + 2\frac{f^{(3)}(x)}{3!}h^3 + \cdots \Rightarrow f'(x_i) = \frac{f(x_i+h)-f(x_i-h)}{2h} + \frac{f^{(3)}(x)}{3!}h^2 + \ldots$, t.y.

$$f'(x_i) = \frac{f(x_i+h)-f(x_i-h)}{2h} + O(h^2), \qquad (1.16)$$

jei $f^{(3)}(x)$ aprėžta.

Centrinis skirtuminis santykis, gautas iš kairinio ir dešininio, naudoja daugiau taškų išvestinės aproksimacijai, bet ir duoda aukštesnės eilės atžvilgiu h paklaidą.



1.3 pav. Paprasčiausių skirtuminių santykių iliustracija

Galima sudaryti aukštesnės tikslumo eilės aproksimacijas imant daugiau narių iš Teiloro eilutės. Pavydžiui, pirmos eilės išvestinės aproksimacija trečiaisiais dešininiais skirtumais, t,y, naudojant tris mazgus į priekį, pradedant nuo x_i imtinai - $f(x_i) = y_i$, $f(x_i + h) = y_{i+1}$ ir $f(x_i + 2h) = y_{i+2}$:

$$f(x_i + h) = f(x_i) + f'(x_i)h + \frac{f''(x_i)}{2!}h^2 + \frac{f^{(3)}(x_i)}{3!}h^3 + \cdots \ldots \quad (1.17)$$

$$f(x_i + 2h) = f(x_i) + f'(x_i)2h + \frac{f''(x_i)}{2!}4h^2 + \frac{f^{(3)}(x_i)}{3!}8h^3 + \cdots \ldots \quad (1.18)$$

Eliminavę $f''(x_i)$ gauname:

$$f'(x_i) = \frac{-3y_i+4y_{i+1}-y_{i+2}}{2h} + \frac{f^{(3)}(x_i)}{3}h^2 + \cdots, \text{ t.y. } f'(x_i) = \frac{-3y_i+4y_{i+1}-y_{i+2}}{2h} + O(h^2), \quad (1.19)$$

jei $f^{(3)}(x)$ aprėžta.

Aukštesnių eilių išvestinės taip pat gali būti aproksimuojamos įvairiais tikslumais atžvilgiu h.

Įvairių tikslumų aproksimacijų skirtuminiais santykiais koeficientus bet kurios eilės išvestinei rasime naudodami neapibrėžtųjų koeficientų metodą. Vieno kintamojo funkcijai f(x) bendru atveju galima užrašyti lygybę:

$$\frac{h^d}{d!}f^{(d)}(x) + O\left(h^{d+p}\right) = \sum_{i=i_{min}}^{i_{max}} C_i f(x + ih) \quad (1.20)$$

Atmetus narį $O\left(h^{d+p}\right)$, kai h – mažas, gaunama reikiama aproksimacija. Čia d - aproksimuojamos išvestinės eilė, $d + p$ - pasirenkama aproksimavimo tikslumo eilė. Priklausomai nuo taškų i_{min}, i_{max} pasirinkimo gaunamos aproksimacijos kairiniais, centriniais ar dešininiais skirtuminiais santykiais. (1.20) formulėje vietoje $f(x + ih)$ įrašę f(x) skleidinį Teiloro eilute:

$$\frac{h^d}{d!}f^{(d)}(x) + O\left(h^{d+p}\right) = \sum_{i=i_{min}}^{i_{max}} C_i \sum_{n=0}^{\infty} i^n \frac{h^n}{n!}f^{(n)}(x) = \sum_{n=0}^{d+p-1}\left(\sum_{i=i_{min}}^{i_{max}} i^n C_i\right)\frac{h^n}{n!}f^{(n)}(x) + O\left(h^{d+p}\right) \quad (1.21)$$

Iš čia:

$$f^{(d)}(x) = \frac{d!}{h^d}\sum_{n=0}^{d+p-1}\left(\sum_{i=i_{min}}^{i_{max}} i^n C_i\right)\frac{h^n}{n!}f^{(n)}(x) + O(h^p) \quad (1.21)$$

Ir kad būtų tenkinama (1.21), turi būti tenkinamos tokios lygybės:

$$\sum_{i=i_{min}}^{i_{max}} i^n C_i = \begin{cases} 0, 0 \le n \le d + p - 1 \text{ ir } n \ne d \\ 1, \ n = d \end{cases} \quad (1.22)$$

Gavome d+p tiesinių lygčių sistemą su $i_{max} - i_{min} + 1$ nežinomųjų. Kad sistema turėtų vienintelį sprendinį, nežinomųjų skaičių prilyginame d+p. Tuomet, kai $i_{min} = 0, i_{max} = d + p - 1$, gaunama

aproksimacija dešininiu skirtuminiu santykiu, kai $i_{min} = -(d + p - 1), i_{max} = 0$ – kairiniu, ir kai $i_{max} = \left[\frac{d+p-1}{2}\right] = -i_{min}$ - centriniu, čia p – lyginis, [ ] žymi skaičiaus sveikąją dalį.

Kelių kintamųjų funkcijoms skirtuminiai santykiai randami analogiškai; šiuo atveju skirtumais aproksimuojamos dalinės išvestinės.

Vieno kintamojo funkcijos d-tosios eilės išvestinės aproksimacija

$$\frac{h^d}{d!} f^{(d)}(x) \approx \sum_{i=i_{min}}^{i_{max}} C_i^{(d)} f(x + ih) \quad (1.23)$$

Tuomet dviejų kintamųjų funkcijai galime du kartus pritaikyti šią aproksimaciją- išilgai x ašies, po to išilgai y ašies:

$$\frac{k^n}{n!}\frac{\partial^n}{\partial y^n}\frac{h^m}{m!}\frac{\partial^m}{\partial y^m} f(x,y) \approx \frac{k^n}{n!}\frac{\partial^n}{\partial y^n}\sum_{i=i_{min}}^{i_{max}} C_i^{(m)} f(x + ih, y) \approx \sum_{i=i_{min}}^{i_{max}}\sum_{j=j_{min}}^{j_{max}} C_i^{(m)} C_j^{(n)} f(x + ih, y + jk) \quad (1.24)$$

Taigi $C_{i,j}^{(m,n)} = C_i^{(m)} C_j^{(n)}$ – dviejų kintamųjų aproksimacijos skirtuminiais santykiais koeficientų matrica yra vektorinė sandauga atitinkamų vieno kintamojo funkcijų aproksimacijų koeficientų vektorių.

Pavyzdžiui, mažiausio tikslumo aproksimacijos skirtuminiais santykiais:

$$f_x(x,y) = \frac{f(x+h,y) - f(x,y)}{h} + O(h) \quad (1.25)$$

$$f_{xx}(x,y) = \frac{f(x+2h,y) - 2f(x+h,y) + f(x,y)}{h^2} + O(h) \quad (1.26)$$

$$f_{xy}(x,y) = \frac{f(x+h,y+k) - f(x+h,y) - f(x,y+k) + f(x,y)}{kh} + O(h) + O(k) \quad (1.27)$$

Teoriškai, didinant išvestinių aproksimavimo tikslumą, t.y. išvestines keičiant aukštesnių eilių skrtuminiais santykiais, galima tikėtis tikslesnių sprendinio aproksimacijų. Taip pat tą galima pasiekti mažinant h, t.y. didinant mazgų skaičių nagrinėjamoje srityje. Pirmasis uždavinio tikslinimo būdas duos vis sudėtingesnes lygčių sistemas, o antrasis – didins sistemos lygčių skaičių, tačiau lygtys bus paprastesnės. Todėl vienas iš uždavinių galėtų būti tyrimas, kaip didėja tikslumas mažinant žingsnį, kaip didėja tikslumas naudojant aukštesnių tikslumo eilių skirtuminius santykius, kada atsiranda didesnės kompiuterinės paklaidos, ar vietoj mažesnio žingsnio paprastesnėje skirtuminėje schemoje galima imti didesnius žingsnius aukštesnių tiklsumo eilių skirtuminėse schemose, ir panašiai.

1.2.3. Skirtuminės schemos sudarymas ir sprendimas

Nagrinėjamoje srityje apibrėžiame tolygųjį diskretųjį tinklą. Tinklas vadinamas tolygiuoju, jei jo žingsnis (atstumas tarp viena kryptimi esančių mazgų) yra pastovus. Tinklo taškai vadinami mazgais. Juose išvestinės pakeičiamos skirtuminiais santykiais (1.20) ir gaunama skirtuminių lygčių sistema, kurios lygčių skaičius lygus mazgų skaičiui. Ją išsprendę, gauname ieškomo sprendinio aproksimacijas mazguose. Radę artinius mazguose, interpoliuodami galėtume rasti sprendinio artinius ir visuose kituose nagrinėjamos srities taškuose.

Taigi kraštinio diferencialinio uždavinio sprendimas suvedamas į lygčių sistemos sprendimą. Kai nagrinėjamas kraštinis uždavinys yra tiesinis ieškomos funkcijos atžvilgiu, tai aproksimavus jį baigtinių skirtumų schema gaunama tiesinių lygčių sistema. Kai sprendžiamas kraštinis uždavinys yra netiesinis, gaunama netiesinių lygčių sistema, kurią literatūroje siūloma spręsti iteraciniais metodais, kur iteracijos vykdomos tol, kol gaunamas sprendinio aproksimacijų pasikeitimas yra norimai mažas, t.y. įvykdyta iš anksto apsibrėžta konvergavimo sąlyga.

Darbe nagrinėsime tiesines baigtinių skirtumų schemas trim atvejais: vieno erdvės kintamojo, dviejų erdvės kintmųjų stacionarūs uždaviniai bei vieno erdvės kintamojo nestacionarus uždavys.

1.3. Kraštiniai diferencialiniai uždaviniai

1.3.1. Paprastųjų diferencialinių lygčių kraštinis uždavinys

Vienmačiu atveju, t.y. paprastajai diferencialinei lygčiai su kraštinėmis sąlygomis intervalo [a; b] galuose, tinklas

$$\left\{ x_0 = a, x_i = x_{i-1} + h, i = 1, \dots, N,\ h = \frac{b-a}{N-1} \right\}; \tag{1.28}$$

čia x_i - mazgai. Tuomet pakeitus sprendinio išvestines baigtiniais skirtuminiais santykiais bus gauta N+1 lygčių sistema - skirtuminė schema, aproksimuojanti vienmatį kraštinį uždavinį.

Programą tyrimui sudarysime tiesinei antros eilės diferencialinei lygčiai bendru pavidalu su pirmojo tipo kraštinėmis sąlygomis;

$$\begin{cases} a_1(x)\frac{d^2u}{dx^2} + a_2(x)\frac{du}{dx} + a_3(x)u + a_4(x) = 0, a < x < b; \\ u(a) = c, u(b) = d. \end{cases} \tag{1.29}$$

modifikavus pirmos ir paskutinės lygčių sudarymą, metodą lengvai modifikuotume kitų tipų kraštinėms sąlygoms.

Kai $a_1(x) = -k(x), a_2(x) = -\frac{dk(x)}{dx}, a_3(x) = q(x), a_4(x) = -f(x)$, gauname šilumos pasiskirstymą strype kuris užima sritį [0, l], aprašantį kraštinį uždavinį:

$$\begin{cases} -\frac{d}{dx}\left(k(x)\frac{du}{dx}\right) + q(x)u = f(x), \ 0 < x < l; \\ u(0) = u_0, \ u(l) = u_l. \end{cases} \quad (1.30)$$

Čia $u(x)$ - strypo temperatūra taške x, $k(x)$ - šilumos laidumo koeficientas, $q(x)$ - paviršiaus šilumos mainų su aplinka koeficientas, $f(x)$ - išorinių šilumos šaltinių tankis, o strypo galų temperatūra palaikoma pastovi.

Tarsime, kad diferencialinio uždavinio koeficientai tenkina eliptiškumo sąlygą:

$$k(x) \geq k_0 > 0, q(x) \geq 0 \quad (1.31)$$

Tuomet yra žinoma, kad diferencialinis uždavinys turi vienintelį sprendinį (5 knyga). Suformuluotas uždavinys turi turėti vienintelį sprendinį, nes jei netenkinama sprendinio vienatis, sprendžiamos sistemos matrica bus išsigimusi.

Sistemą aproksimavę mažiausio tikslumo skirtuminiais santykiais, gautume tiesinių algebrinių lygčių sistemą su trijstrižaine matrica:

$$\begin{pmatrix} 1 & & & & & & \\ A_1 & B_1 & C_1 & & & O & \\ & A_2 & B_2 & C_2 & & & \\ & & & \ddots & & & \\ & & & & A_{N-1} & B_{N-1} & C_{N-1} \\ & O & & & & & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix} = \begin{pmatrix} c \\ -a_4(x_1) \\ -a_4(x_2) \\ \vdots \\ -a_4(x_{N-1}) \\ d \end{pmatrix} \quad (1.32)$$

Arba trumpiau $A\hat{U} = F$.

Čia A_i, B_i, C_i - koeficientai, priklausantys nuo skirtuminių santykių koeficientų (mažiausias tikslumas antros eilės diferencialinei lygčiai yra $O(h^2)$, tad tai būtų centriniai skirtuminiai santykiai per 3 taškus) bei funkcijų reikšmių tinklelio mazguose $a_j(x_i)$, $j = \overline{1:3}$ bei tinklelio žingsnio h. Matome, kad pirmą ir paskutinę eilutę būtų galima lengvai eliminuoti, tačiau išlaikydami bendrumą to nedarysime, nes tokiu būdu metodas bus lengviau pritaikomas bet kokio tipo kraštinėms sąlygoms, ir, be to, lengviau suprantamas bei realizuojamas.

Metodą baigtinių skirtumų schemai sudaryti naudojant bet kurios eilės tikslumo skirtuminius santykius aprašysime tiriamojoje dalyje.

1.3.2. Lokalioji ir globalioji paklaidos, stabilumas

Kai artiniais pakeičiame išvestines visuose tinklelio mazguose, gauname sistemą, siejančią visus nagrinėjamus taškus, ir galime tikėtis, kad ir gautas sprendinys mazguose bus tos pačios eilės tikslumo. Paklaidos įvertinimui naudosime maksimumo normą:

$$\|V\|_\infty = \max_j |V_j| \tag{1.33}$$

Ši norma yra patogi tuo, kad palyginus su kitomis dažnai sutinkamomis normomis ji yra stipriausia: jei nors vienas elementas bus mažesnės tikslumo eilės, tai ir norma bus tos eilės.

Tegu $U_i = u(x_i)$ – tikrojo sprendinio reikšmės tinklelio mazguose, $\widehat{U} = (\hat{u}_0, \hat{u}_1, \dots, \hat{u}_N)^T$- baigtinių skirtumų metodu gautos sprendinio aproksimacijos to paties tinklelio mazguose. Tuomet absoliutinė paklaida kiekviename tinklelio mazge: $E = U - \widehat{U}$. Tai globalioji paklaida – lyginamas tikrasis sprendinys su gauta jo aproksimacija.

Reikia parodyti, kad $\|E\|_\infty$ yra tos pačios eilės tikslumo, kaip ir sprendinio išvestinės aproksimacija kiekviename taške, tuomet būtų aišku, kad gauto sprendinio aproksimacijos tikslumo eilė yra nemažesnė. Šis tyrimas grindžiamas tokia schema, kuri dažnai vadinama fundamentaliąją baigtinių skirtumų metodų teorema:

$$O(h^p) \text{ lokalioji paklaida } + \text{stabilumas } => O(h^p) \text{ globalioji paklaida} \tag{1.34}$$

Kiekviename tinklelio mazge sprendinio išvestinę pakeisdami jos artiniu žinome, kokios eilės narį atmetame, taigi lokaliąją paklaidą žinome jau tuomet, kai renkamės schemos tikslumą: $\tau_j = O(h^p)$, o lokaliųjų paklaidų vektorių nagrinėjamai schemai galime užrašyti per skirtuminės schemos matricą, gautą pasirinkus tinklelio žingsnį h: $\tau = AU - F$. Iš šios lygties, ir iš lygties $A\widehat{U} = F$, gauname lokaliąją ir globaliąją paklaidas siejančią lygtį: $AE = -\tau$ su kraštinėmis sąlygomis $E_0 = 0, E_N = 0$ (nes kraštinės sąlygos yra pirmojo tipo, taigi jų neaproksimavome).

Akcentuodami, kad gautoje lygtyje elementai priklauso nuo pasirinkto žingsnio h, užrašysime:

$$A^h E^h = -\tau^h \tag{1.35}$$

Sistemos sprendinys (matrica A neišsigimusi, nes tai ta pati matrica, kaip ir pagrindinio uždavinio, o tarėme, kad uždavinys suformuluotas korektiškai):

$$E^h = -\left(A^h\right)^{-1}\tau^h \tag{1.36}$$

$$\left\|E^h\right\| = \left\|\left(A^h\right)^{-1}\tau^h\right\| \le \left\|\left(A^h\right)^{-1}\right\| \left\|\tau^h\right\| \tag{1.37}$$

Žinome, kad $\left\|\tau^h\right\|_\infty = O(h^p)$, jei visuose mazguose sprendinio išvestines aproksimavome mažiausiu tikslumu $O(h^p)$. Matome, kad tokios pačios tikslumo eilės galime tikėtis ir globaliajai paklaidai, jeigu $\left\|\left(A^h\right)^{-1}\right\| \le C, \forall h < h_0$. (1.38)

Sakoma, kad metodas yra stabilus, jei $\left(A^h\right)^{-1}$ egzistuoja ir $\left\|\left(A^h\right)^{-1}\right\|$ yra aprėžta visiems pakankamai mažiems tinklelio žingsniams h.

Šis teiginys galioja visiems tiesiniams kraštiniams uždaviniams.

Sakoma, kad metodas yra suderinamas (angl. consistent) su kraštiniu uždaviniu, jeigu

$\left\|\tau^h\right\| \underset{h\to 0}{\longrightarrow} 0$ (1.39)

Jei pasirenkame sprendinio išvestinių aproksimacijas naudodamiesi skirtuminiais santykiais, tai metodas bus suderinamas su uždaviniu, nes $\left\|\tau^h\right\|_\infty = O(h^p)$, p-pasirenkamas sudarant skirtuminę schemą.

Sakoma, kad metodas konverguoja, jei

$\left\|E^h\right\| \underset{h\to 0}{\longrightarrow} 0$ (1.40)

Konvergavimą apibūdina schema:

$\text{suderinamumas} + \text{stabilumas} => konvergavimas$ (1.41)

Iš tikrųjų, $\left\|E^h\right\| \le \left\|\left(A^h\right)^{-1}\right\| \left\|\tau^h\right\| \le C\left\|\tau^h\right\| \underset{h\to 0}{\longrightarrow} 0$. Ši schema yra analogiška (1.34).

Schemos stabilumą patikrinti teoriškai net tiesiniams uždaviniams yra sunku. Literatūroje stabilumas tiriamas mažų tikslumo eilių schemoms, imant sukonkretintą uždavinį. Pavydžiui (5 knyga), naudojantis maksimumo principu (absoliutine verte didžiausia paklaida įgyjama tinklelio vidiniame taške) išnagrinėta (1.30) uždavinio schema, sudaryta naudojant $O(h^2)$ tikslumo skirtuminius santykius. Gauta, kad esant tenkinamoms eliptiškumo sąlygoms, naudojant maksimumo normą, schema yra stabilioji, taigi globalioji paklaida yra $O(h^2)$ eilės. Kraštiniams uždaviniams su kitų tipų kraštinėmis sąlygomis stabilumą įrodyti yra dar sunkiau.

Šiame darbe teoriškai stabilumo nevertinsime, nes tiriama didelė skirtuminių schemų įvairovė, ir didelė dalis jų yra didelių tikslumo eilių. Neturint galimybės teoriškai įrodyti schemos stabilumą, literatūroje (1 knyga) siūloma patikrinti, ar nagrinėjmos schemos skaičiavimų rezultatai duoda teoriškai tikėtiną konvergavimo greitį. Taip pat stabilumą maksimumo normos prasme galima ištirti skaitiškai, tikrinant sąlygą (1.38), t.y. skaičiuojant atvirkštinių matricų maksimumo normų seką (1 knyga) – tam net nereikia turėti tikslaus sprendinio, tad toks tyrimas gali būti atliktas ir sprendžiant realų tiesinį (ar ištiesintą) uždavinį. Matricos normą skaičiuosime tokiu būdu:

$\|A\|_\infty = \max_{1\le i\le m} \sum_{j=1}^{m} \left|a_{ij}\right|$ (didžiausia eilutės suma) (1.42)

Aproksimuojant kraštinį uždavinį esant antros ar trečios eilės kraštinėms sąlygoms jas reikia aproksimuoti ne žemesnės eilės skirtumais nei išvestines diferencialinėje lygtyje. Geriausia išvestines viename uždavinyje aproksimuoti tos pačios eilės paklaidas duodančiais skirtumais, nes kaip matėme, stabiliose schemose aproksimacijos paklaidos eilė priklauso nuo visų dedamųjų paklaidų eilių.

Imant aukštesnio tikslumo aproksimacijas, sudarant lygtis arti srities kraštų iškyla problema, kad nėra apibrėžta pakankamai mazgų. Tokiu atveju naudoti mažesnio tikslumo aproksimacijas yra negerai, nes atitinkamai iki mažiausios eilės sumažės viso uždavinio sprendinio tikslumo eilė – nebus gautas siekiamas tikslumas, nors uždavinys sprendžiamas sudėtingesnis. Kad būtų gauta maksimali schemos aproksimuojamo uždavinio tikslumo eilė, kraštinės sąlygos, jei jos yra ne pirmojo tipo, ir visi vidiniai tinklelio taškai turi būti aproksimuojami tos pačios eilės tikslumu.

1.3.3. Ekstrapoliacija

5-toje knygoje nagrinėjant vienmatį tiesinį kraštinį uždavinį pasiūlyti keli būdai $O(h^4)$ tikslumo sprendiniams rasti. Pirmas būdas – jau minėta skirtuminė schema, aproksimuojant $O(h^4)$ tikslumo skirtuminiais santykiais, o antras būdas – pavadintas ekstrapoliacija: išspręsti uždavinį naudojant $O(h^2)$ tikslumo skirtumines schemas du kartus - žingsniais h ir h/2; tuomet žingsnio h tinklelyje ekstrapoliuoti sprendinį, panaikinant žemiausios eilės paklaidos dedamąją ir tokiu būdu gauti sprendinį $O(h^4)$ tikslumu.

Didesnio žingsnio tinklelyje:

$$U_j \approx u(jh), i = 1,2, \dots, m \tag{1.43}$$

Mažesnio žingsnio tinklelyje:

$$V_i \approx u(ih/2), i = 1,2, \dots, 2m + 1 \tag{1.44}$$

Čia U_j ir V_{2j} aproksimuoja $u(jh)$.

Kadangi U_j ir V_{2j} apskaičiuoti $O(h^2)$ tikslumu, tai paklaidos

$$U_j - u(jh) = C_2 h^2 + C_4 h^4 + C_6 h^6 + \cdots \tag{1.45}$$

$$V_{2j} - u(jh) = C_2 \left(\frac{h}{2}\right)^2 + C_4 \left(\frac{h}{2}\right)^4 + C_6 \left(\frac{h}{2}\right)^6 + \cdots \tag{1.46}$$

Koeficientai $C_2, C_4, \dots$ priklauso nuo aukštesnių eilių išvestinių, bet neprikalauso nuo h kiekviename fiksuotame taške jh. Tuomet ekstrapoliuotas sprendinys žingsnio h tinklelyje:

$$\widehat{U}_j = \frac{1}{3}\left(4V_{2j} - U_j\right) + O(h^4) \tag{1.47}$$

Galima ekstrapoliuoti ir toliau, uždavinį sprendžiant žingsniu h/4 – gaunama 6-os eilės aproksimacija, ir t.t. Šią idėją išplėsime ir panaudosime tyrime.

1.3.4. Skaitinių sprendinių paklaidų įvertinimas

Tyrime nagrinėsime tik uždavinius, kurių tikslius sprendinius žinome. Tuomet galėsime tiksliai įvertinti, kaip asimptotiškai keičiasi paklaidos, smulkinant žingsnį arba didinant aproksimavimo tikslumo eilę, įvertinti konvergavimo greitį, palyginti jį su teoriniu. Tegu h žingsnio tinklelyje spręsto uždavinio paklaida lyginama su tiksliu sprendiniu to tinklelio mazguose, ir apibrėžiama kaip norma:

$$P(h) = \left\| U^h - \widehat{U}^h \right\|. \tag{1.48}$$

Jei skirtuminė schema yra p-tosios tikslumo eilės, tai galima tikėtis, kad

$$P(h) = Ch^p + o(h^p) \tag{1.49}$$

ir jei h pakankamai mažas, tai

$$P(h) \approx Ch^p \tag{1.50}$$

Išsprendę uždavinį žingsniais h_1 ir h_2, gautume $\frac{P(h_1)}{P(h_2)} \approx \left(\frac{h_1}{h_2}\right)^p$, ir iš čia tikslumo eilės įvertis

$$p \approx \frac{\ln(P(h_1)/P(h_2))}{\ln(h_1/h_2)}. \tag{1.51}$$

Tikslumo eilę galima įvertinti ir mažiausių kvadratų metodu:

$$\ln\big(P(H)\big) = \ln C + p\ln(H) \tag{1.52}$$

Čia H – skirtingų tinklelio žingsnių vektorius. Reikia išspręsti tiesinę lygčių sistemą:

$$\begin{pmatrix} 1 & \ln(h_1) \\ 1 & \ln(h_2) \\ \vdots & \vdots \\ 1 & \ln(h_k) \end{pmatrix} \begin{pmatrix} \ln C \\ p \end{pmatrix} = \begin{pmatrix} \ln\big(P(h_1)\big) \\ \ln\big(P(h_2)\big) \\ \vdots \\ \ln\big(P(h_k)\big) \end{pmatrix} \tag{1.53}$$

Kai neturima tikslaus sprendinio, paklaidas galima įvertinti lyginant sprendinio įverčius, apskaičiuotus naudojant skirtingų žingsnių tinklelius.

1.3.5. Dvimatis stacionarus kraštinis uždavinys

Nors programą nesunku parašyti elipsiniam kraštiniam uždaviniui bendru atveju, įtraukiant kintamus koeficientus, nagrinėjimui pasirinksime atskirą atvejį - Puasono diferencialinę lygtį su pirmojo tipo kraštinėmis sąlygomis:

$$\begin{cases} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), (x,y) \in \Omega; \\ u(x,y) = u_0(x,y), \ (x,y) \in \partial\Omega. \end{cases} \tag{1.54}$$

Ši diferencialinė lygtis aprašo šilumos pasiskirstymą paviršiuje, kai kraštuose temperatūra yra palaikoma pastovi ir yra nusakyta kraštinėmis sąlygomis, o stacionarių šilumos šaltinių padėtis aprašoma funkcija f.

Eliptiškumo sąlyga Puasono lygčiai tenkinama visoje srityje. Tarsime, kad uždavinys suformuluotas korektiškai, ir jam egzistuoja vienintelis sprendinys $u \in C^{4,4}(\Omega)$.

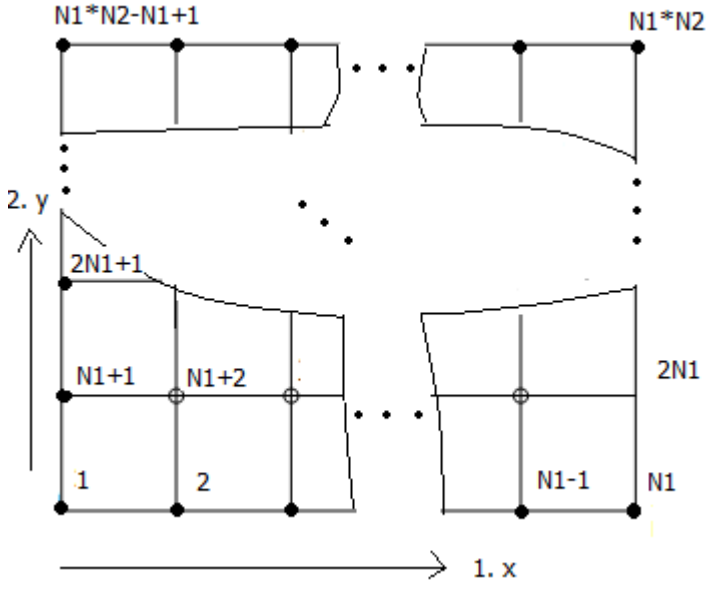
Nagrinėjimui imsime vienetinį kvadratą (nors programą rašysime bendresniu atveju - stačiakampiui) $\Omega = \{(x, y): 0 \le x \le 1, 0 \le y \le 1\}$.

Spręsdami uždavinį baigtinių skirtumų metodu, pirmiausiai srityje Ω apibrėžiame diskretųjį tinklą:

$$\left\{ (x_i, y_j): x_i = ih, y_j = jk,\ i = 0, \dots, N_1 - 1, j = 0, \dots, N_2 - 1, , h = \frac{1}{N_1 - 1}, k = \frac{1}{N_2 - 1} \right\}; \qquad (1.55)$$

čia (x_i, y_j) – tinklelio mazgai.

Gauto tinklelio mazgus reikia sunumeruoti tam tikra pasirinkta tvarka; toliau naudosime tokią numeraciją:



1.4 pav. 2D tinklelio mazgų numeracija

Kraštiniame diferencialiniame uždavinyje pakeitus sprendinio išvestines skirtuminiais santykiais gaunamos $N_1 N_2$ lygčių (bendrumo dėlei įtraukiami ir krašto taškai, tuomet algoritmas lengvai modifikuojamas ir uždaviniams su kitų tipų kraštinėmis sąlygomis) - skirtuminė schema, aproksimuojanti dvimatį Puasono kraštinį uždavinį.

Gautos algebrinės lygtys mazgų numeracijos eilės tvarka surašomos į lygčių sistemą matricinėje formoje.

Sistemą aproksimavus mažiausio tikslumo centriniais skirtuminiais santykiais, gaunama:

$$\begin{cases} \frac{u_{i-1,j}-2u_{i,j}+u_{i+1,j}}{h^2}+\frac{u_{i,j-1}-2u_{i,j}+u_{i,j+1}}{k^2}=f_{i,j},\ i=1,\dots,N_1-2, j=1,\dots,N_2-2 \\ u_{0,j}=g(0,y_j),\ u_{N_1-1,j}=g(1,y_j), j=0,\dots,N_2-1,\ u_{i,0}=g(x_i,0), u_{i,N_2-1}=g(x_i,1) \end{cases} \quad (1.56)$$

Parinkus vienodus žingsnius h=k, t.y. $N=N_1=N_2$, schema supaprastėja. Gaunama sistema su tokia trijstrižaine simetrine matrica:

$$\begin{pmatrix} I_N & & & 0 & & \\ I_N^1 & T_N^1 & I_N^1 & & & \\ & I_N^1 & T_N^1 & I_N^1 & & \\ & & & \ddots & & \\ & & & I_N^1 & T_N^1 & I_N^1 \\ & 0 & & & & I_N \end{pmatrix} \begin{pmatrix} u_0 \\ u_2 \\ \vdots \\ u_{N-2} \\ u_{N-1} \end{pmatrix} = \begin{pmatrix} u_{-,0} \\ f_1^1 \\ \vdots \\ f_{N-2}^1 \\ u_{-,N-1} \end{pmatrix} \quad (1.58)$$

čia pažymėjome $(N)*(N)$ blokelius:

$$T_N^1 = \begin{pmatrix} 1 & & & & 0 & \\ 1 & -4 & 1 & & & \\ & 1 & -4 & 1 & & \\ & & & & & \\ & & & 1 & -4 & 1 \\ & 0 & & & & 1 \end{pmatrix}$$

$$I_N^1 = \begin{pmatrix} 0 & & & & & \\ & 1 & & & & 0 \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & \\ & 0 & & & & 0 \end{pmatrix}, I_N = \begin{pmatrix} 1 & & & & & \\ & 1 & & & & 0 \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & \\ & 0 & & & & 1 \end{pmatrix}$$

$u_i = (u_{0,i}, \dots, u_{N-1,i})^T$, $f_i^1 = (g_{0,i} f_{1,i}, \dots, f_{N-1,i}, g_{N-1,i})^T$, $i=0,\dots,N-1$;

$u_{-,0}$ $u_{-,N-1}$ – sprendinio reikšmės apatiniame ir viršutiniame kraštuose, o vektoriai f apima kraštines reikšmes srities kairėje ir dešinėje.

Sistemą galima užrašyti glausčiau, kraštinių tinklelio taškų neįtraukiant į sistemą:

$$\begin{pmatrix} T_{N-2} & I_{N-2} & \dots & 0 \\ I_{N-2} & T_{N-2} & & \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & \begin{matrix} T_{N-2} & I_{N-2} \\ I_{N-2} & T_{N-2} \end{matrix} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-2} \end{pmatrix} = \begin{pmatrix} -b_1+h^2 f_1 \\ -b_2+h^2 f_2 \\ \vdots \\ -b_{N-2}+h^2 f_{N-2} \end{pmatrix} \quad (1.59)$$

čia

$T_{N-2} = \begin{pmatrix} -4 & 1 & & & \\ 1 & -4 & 1 & & 0 \\ & 1 & -4 & & \\ & & & -4 & 1 \\ & 0 & & 1 & -4 \end{pmatrix}$ - triįstrižainė mtrica, aproksimuojanti išilgai tinklelio eilutės, dimensija $(N-2)*(N-2)$; $u_i = \left(u_{1,i}, \dots, u_{N-2,i}\right)^T$, $f_i = \left(f_{1,i}, \dots, f_{N-2,i}\right)^T$, $i = 1, \dots, N-2$, $b_1 = \left(u_{0,1} + u_{1,0}, u_{2,0}, \dots, u_{N-3,0}, u_{N-2,0} + u_{N-1,1}\right)^T$, $b_i = \left(u_{0,i}, 0, \dots 0, u_{N-1,1}\right)^T, i = 2, \dots, N-3$ ir $b_{N-2} = \left(u_{0,N-2} + u_{1,N-1}, u_{2,N-1}, \dots, u_{N-3,N-2}, u_{N-2,N-1} + u_{N-1,N-2}\right)^T$

Čia sistemos sprendinys - vidiniai tinklelio taškai; krašto taškų svoriai perkelti į laisvą stulpelį. Tą naudinga padaryti, nagrinėjant šią paprasčiausią schemą, nes gaunama mažesnės dimensijos sistema. Tačiau tyrime aprašysime bendrą metodą pasirinktai tikslumo eilei, kuris būtų lengvai modifikuojamas bet kurio tipo kraštinėms sąlygoms, todėl krašto taškų iš sistemos neeliminuosime, kaip ir vienmačiame uždavinyje.

Stabilumas 2D stacionariu tiesiniu atveju gali būti tiriamas naudojantis tuo pačiu metodu, kaip ir jau nagrinėtu 1D atveju – nagrinėjant $\left\| \left(A^h\right)^{-1} \right\|$ aprėžtumą mažėjant tinklelio žingsniui h.

Literatūroje paprastai apsiribojama mažiausių tikslumų schemų analize. Pvz. 5 knygoje stabilumas išnagrinėtas Puasono dvimačiam kraštiniam uždaviniui remiantis maksimumo principu (skirtuminėje schemoje esant tenkinamoms tam tikromis sąlygomis apie matrices A koeficientus, sprendinys negali įgyti ekstremalių reikšmių ne tinklelio taškuose) – tokia schema yra stabili ir jos tikslumo eilė yra 2-oji.

1.3.6. Vienmatis nestacionarus kraštinis uždavinys

Nagrinėjimui pasirinksime atskirą pradinio-kraštinio uždavinio atvejį - parabolinę diferencialinę lygtį su pirmojo tipo kraštinėmis sąlygomis ir pradine sąlyga:

$$\begin{cases} \frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \ 0 < x < L, t > 0; \\ u(0,t) = u_0, \ u(L,t) = u_L, \ u(x,0) = f_0(x). \end{cases} \quad (1.60)$$

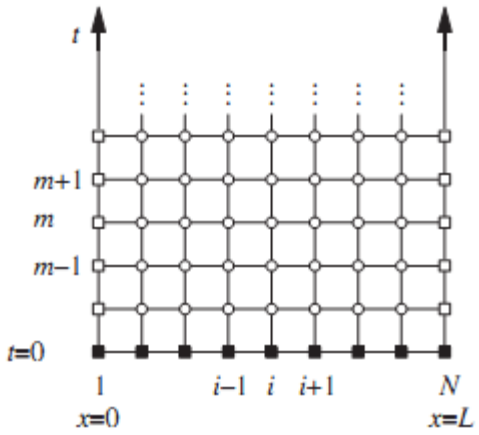
Ši diferencialinė lygtis aprašo pereinamąjį procesą šilumos pasiskirstymo sritį [0, L] užimančiame strype, kai jo galuose temperatūra yra palaikoma pastovi, kaip nusakyta kraštinėmis sąlygomis, ir nėra išorinių šilumos šaltinių. Pradiniu laiko momentu $u(x,0) = f_0(x)$.

Spręsdami uždavinį baigtinių skirtumų metodu, pirmiausiai srityje $[0, L] * [0, t_{max}]$ apibrėžiame diskretųjį tinklą:

$$\left\{ \left(x_i, t_j\right): x_i = ih, t_j = jk, \ i = 0, \dots, N-1, j = 0, \dots, M-1, , h = \frac{L}{N-1}, k = \frac{t_{max}}{M-1} \right\}; \quad (1.61)$$

čia (x_i, t_j) – tinklelio mazgai.

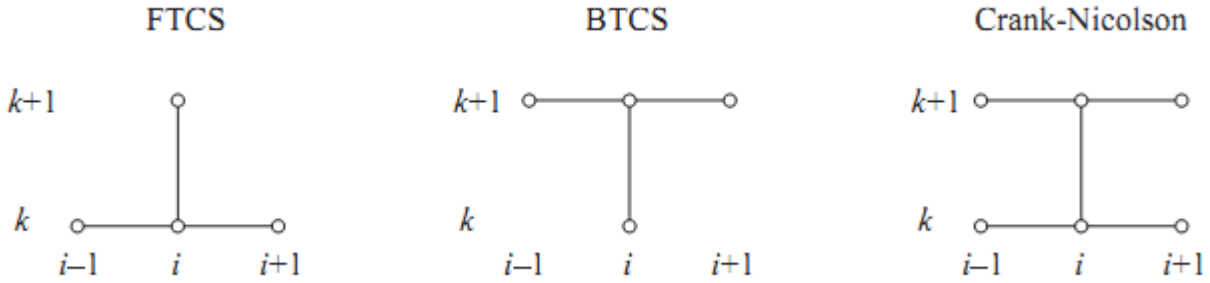
Tinklelio schemoje apskritimai žymi vidinius tinklelio mazgus, kuriuose reikia rasti sprendinio aproksimacijas, užtušuoti kvadratai žymi pradines sąlygas, o neužtušuoti kvadratai – kraštines sąlygas.



1.5 pav. 1Dt tinklelis

Pradiniame-kraštiniame diferencialiniame uždavinyje pakeitus sprendinio išvestines skirtuminiis santykiais, priklausomai nuo pasirinktų skirtuminų santykių – gaunama skirtuminė schema, aproksimuojanti šį uždavinį.

Literatūroje dažniausiai nagrinėjamos tokios aproksimavimo schemos:



1.6 pav. Paprasčiausios 1Dt uždavinio aproksimavimo schemos

Dešininė laiko – centrinė erdvės schema (angl. FTCS) – išreikštinė schema:

$$\frac{u_i^{m+1}-u_i^m}{k} = \alpha\frac{u_{i-1}^m-2u_i^m+u_{i+1}^m}{h^2} + O(k) + O(h^2) \qquad (1.62)$$

$$u_i^{m+1} = u_i^m + \frac{\alpha k}{h^2}(u_{i+1}^m - 2u_i^m + u_{i-1}^m) \qquad (1.63)$$

Naudojantis Niumano (angl. Neumann) stabilumo analize parodoma, kad ši schema yra stabili, kai $r = \frac{\alpha k}{h^2} < \frac{1}{2}$. Nagrinėjamos lygties sprendiniai, kai kraštinės sąlygos yra konstantos, yra gęstnčios

funkcijos, t.y. sprendinys laikui bėgant kis nuo pradinės sąlygos iki konstantos - kol nusistovės. Todėl kai r>1/2, t.y. psirinktas laiko žingsnis tinklelyje sąlyginai (lyginant su h) didelis, galima tikėtis, jog gauti sprendiniai osciliuos ar augs dėl metodo nestbilumo. Šio metodo privalumas tas, kad jis yra išreikštinis, ir nereikia spręsti lygčių sistemos, o sprendinys nujame žingsnyje apskaičiuojamas tiesiogiai:

$$u^{m+1} = Au^m \tag{1.64}$$

čia

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ r & (1-2r) & r & 0 & 0 & 0 \\ 0 & r & (1-2r) & r & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & r & (1-2r) & r \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Kairinė laiko- centrinė erdvės schema (angl. BTCS) – visiškai neišreikštinė schema:

$$\frac{u_i^m - u_i^{m-1}}{k} = \alpha \frac{u_{i-1}^m - 2u_i^m + u_{i+1}^m}{h^2} + O(k) + O(h^2). \tag{1.65}$$

Šiuo atveju kiekviename laiko žingsnyje reikia spręsti lygčių sistemą, nes u_i^m priklauso nuo šalia esančių mazgų reikšmių tuo pačiu laiko momentu m. Taigi galima sakyti, kad sprendžiama M-1 kraštinių vienmačių stacionarių uždavinių, į schemos matricą kiekviename žingsnyje įtraukiant praeities reikšmę u_i^{m-1}.

Aproksimavimo paklaidos yra tos pačios eilės kaip ir FTCS schemoje, o uždavinys sprendžiamas sudėtingiau. Tačiau įrodoma, kad ši schema nagrinėjamam uždaviniui yra besąlygiškai stabili.

Kranko Nikolsono (Crank Nicolson) schema.

Ši schema, lyginant su ankstesnėmis, yra aukštesnės eilės tikslumo laiko atžvilgiu. Tai taip pat neišreikštinė schema. Išvestinė erdvės kintamojo atžvilgiu yra aproksimuojama vidurkiu centrinių aproksimacijų esamo ir buvusio laiko žingsnio, o laikas aproksimuojams kairiniu skirtuminiu santykiu:

$$\frac{u_i^m - u_i^{m-1}}{k} = \frac{\alpha}{2}\left(\frac{u_{i-1}^m - 2u_i^m + u_{i+1}^m}{h^2} + \frac{u_{i-1}^{m-1} - 2u_i^{m-1} + u_{i+1}^{m-1}}{h^2}\right) + O(k^2) + O(h^2). \tag{1.66}$$

Algoritmiškai Kranko-Nikolsono schema yra analogiška BTCS shemai, bet duoda tikslesnius laiko atžvilgiu rezultatus, be to, parodoma, kad ji taip pat besąlygiškai stabili. Tačiau tam, kad gautume dar aukštesnės nei antros eilės laiko atžvilgiu aproksimaciją, reikia ieškoti kitų schemų. Todėl šio darbo tyrimui pasirinkome BTCS schemą, ją modifikuosime aukštesnių eilių aproksimacijoms.

Stabilumą tirti teoriškai yra sudėtinga, kai schema yra aukštesnės eilės tikslumo. Tačiau šiam uždaviniui stabilumą galėsime įvertinti atsižvelgę į sprendinio savybes – esant pastovioms kraštinėms sąlygoms, sprendiniai slopsta nuo pradinės sąlygos iki tam tikros pusiausvyros būsenos – nusistovi. Pereinamasis procesas yra tolydus ir aprėžtas – sprendinys neįgyja reikšmių, esančių už pradinio

sprendinio ribų. Tokiu būdu galėsime vizualiai patikrinti naujų – aukštesnių eilių – schemų stabilumą, nes antros eilės schema yra teoriškai besąlygiškai stabili, o aukštesnių eilių schemų stabilumas teoriškai neištirtas.

1.3.7. Paklaidų eilės patvirtinimas praktiškai

$O(h^p)$ nusako kokiu greičiu paklaida atžvilgiu h nyks, kai h mažės. Sakykime, norime įsitikinti, jog sukurta ir aprašyta schema duoda teoriškai tikimąsi konvergavimo greitį.

Tegu tikroji paklaida, gauta išsprendus konkretų uždavinį su konkrečia schema, kai k->0, h->0: $TP = K_t k + K_x h^2$. K_t, K_h- konstantos, priklausančios nuo sprendžiamo uždavinio bei parinktos schemos. Kad TP->0, k,h->0

Kad patikrintumėme, ar aprašyta schema atitinka teorinius rezultatus, galime patikrinti, ar, šiuo atveju, laiko žingsnio k tiesinis pakytis duoda tiesinį paklaidos pokytį, ir ar tiesinis žingsnio h pokytis duoda kvadratinį paklaidos pokytį. Tam fiksuojame vieną žingsnį ir keičiame kitą bei vertiname paklaidos pasikeitimą.

Sakykime, gauti du skaitiniai sprendiniai: $TP_1 = K_t k_1 + K_x h_1^2$ ir $TP_2 = K_t k_2 + K_x h_2^2$.

$$\frac{TP_2}{TP_1} = \frac{K_t k_2 + K_x h_2^2}{K_t k_1 + K_x h_1^2} \tag{1.67}$$

Kad įvertintumėm TP priklausomybę nuo k, fiksuojame mažą žingsnį h ir mažiname k. Jei h yra pakankamai mažas, tai $K_x h \ll K_t k_1$ ir $K_x h \ll K_t k_2$, todėl

$$\left(\frac{TP_2}{TP_1}\right)_{h=const} \approx \frac{K_t k_2}{K_t k_1} = \frac{k_1}{k_2} \tag{1.68}$$

Paklaida TP vertinama kaip norma skirtumo tarp tikrojo ir aproksimuoto sprendinio

Patogu dirbti *Matlab* aplinkoje, nes ji orientuota matriciniams uždaviniams spręsti, patogi vizualizacija.

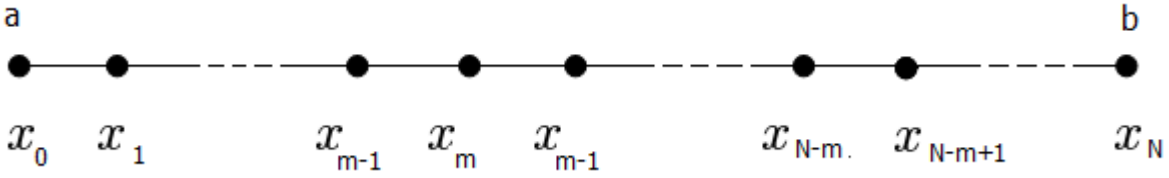
2. TIRIAMOJI DALIS IR REZULTATAI

2.1. Schemų sudarymas

Aprašysime metodą, kuriuo programoje sudaroma matrica bendru atveju. Nagrinėkime lygtį (1.29). Joje išvestines pakeiskime pasirinkto tikslumo $O(h^{2m})$ skirtuminiais santykiais (tikslumo eilė p=2m, nes antra išvestinė gali būti aproksimuojama tik lyginės eilės tikslumo skirtuminiais santykiais):

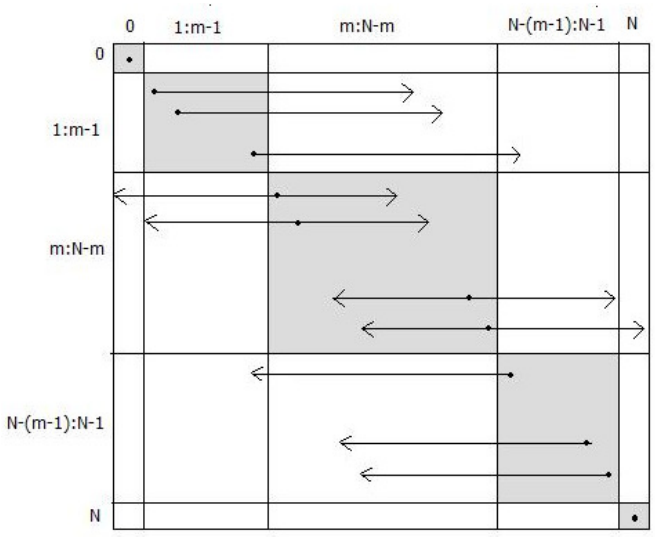
$$\begin{cases} a_1(x_i)u''_{O(h^{2m})}(x_i) + a_2(x_i)u'_{O(h^{2m})}(x_i) + a_3(x_i)u(x_i) + a_4(x_i) = 0, \ 1 \le i \le N-1 \\ u(x_0) = c, u(x_N) = d \end{cases} \quad (2.1)$$

$u''_{O(h^{2m})}(x_i)$ ir $u'_{O(h^{2m})}(x_i)$ - aproksimacijos pagal (1.23) formulę. Vidiniuose tinklelio taškuose užtenka taškų į abi puses aproksimuoti centriniais skirtuminiais santykiais, kai $i = \overline{m, N-m}$



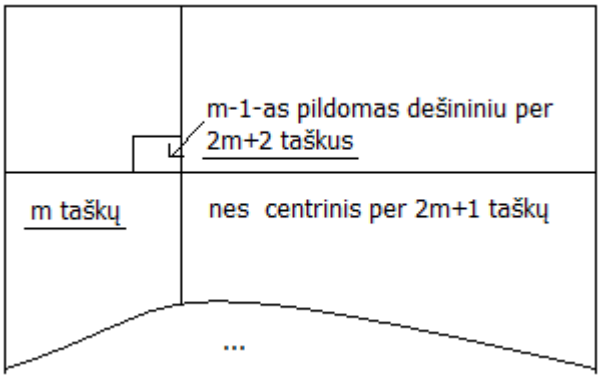
2.1 pav. 1D tinklelis

Taškuose, kur neužtenka mazgų centriniams skirtuminiams santykiams, imame dešininius (tinklelio pradžioje) arba kairinius (tinklelio gale) skirtuminius santykius. Svarbiausia laikytis principo, kad visos aproksimacijos turi būti atliktos tuo pačiu tikslumu. Antroji išvestinė kraštuose aproksimuojama per 2m+2 taškus (kairinis, dešininis skirtuminiai santykiai) Schematiškai gauta tiesinių lygčių sistema pavaizduota 2.2. paveikle.



2.2 pav. 1D uždavinio matricos sudarymas bendru atveju

Norint apskaičiuoti $O(h^{2m})$ tikslumu, reikia mažiausiai 3m+1 dalinimo taškų: 2-os eilės išvestinę aproksimuojančiam centriniam skirtuminiam santykiui pirmiems m taškų centrinio skirtuminio santykio skaičiuoti negalime, skaičiuojamas dešininis per 2m+2 taškus. Dešininiais reikia aproksimuoti iki m-1- ojo taško imtinai, taigi bus mažiausiai (m-1) + (2m+2) stulpelių, t.y. mažiausiai 3m+1 tinkelio taškų.

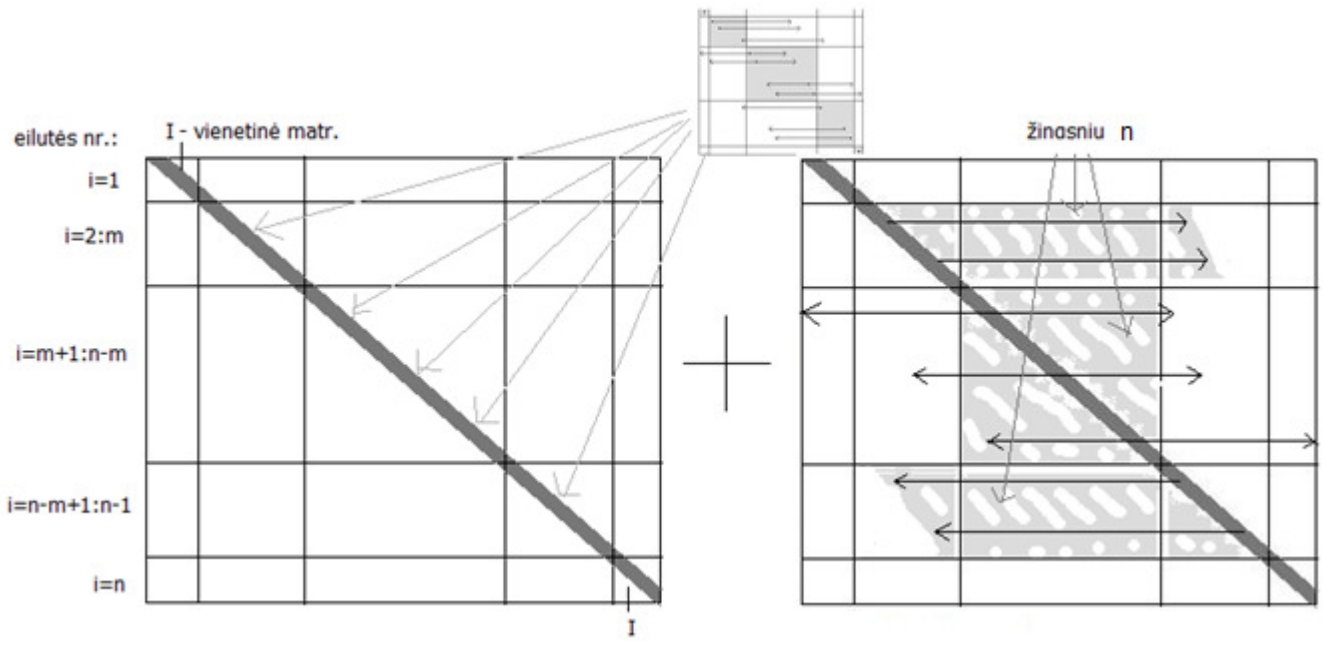


2.3 pav. Minimalaus 1D tinklelio mazgų skaičiaus paieška

Dvimačiu stacionariu atveju skirtuminės schemos lygčių sistemos matrica sudaroma analogiškai: taškuose, kur neužtenka mazgų centriniams skirtuminiams santykiams, imame dešininius (tinklelio pradžioje) arba kairinius (tinklelio gale) skirtuminius santykius.

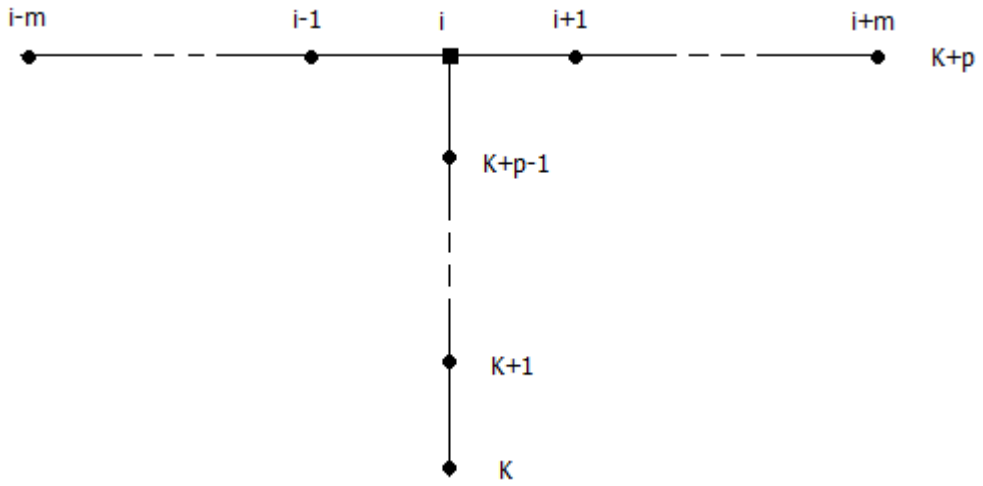
Kai didiname schemos aproksimavimo tikslumą, kaip ir vienmačiu atveju, arti kraštų centriniams skirtuminiams santykiams neužteks esamų taškų, todėl reikės aproksimuoti kairiniais ar dešininiais skirtuminiais santykiais. Didinant tikslumo eilę matrica taps vis sudėtingesnė, nebe trijstrižainė, nes reikės kairinių ir dešininių aproksimacijų. Jei esant mažiausiam tikslumui galima taikyti greitesnius algoritmus sistemos sprendimui, taupiau apraštyti matricą, tai didinant schemų tikslumą prarandami šie privalumai.

Sudarydami matricą dvimačio stacionaraus uždavinio sistemai, iš pradžių užpildome reikiamus koeficientus, eidami išilgai eilučių - atliekame N*N mazgų dalinių aproksimacijų, t.y. aproksimuojame išvestinę $\frac{\partial^2 u}{\partial x^2}$. Šiuo atveju visose tinklelio eilutėse reikės vienodai kairinių, dešininių bei centrinių aproksimacijų, taigi matricoje įstrižainė užpildoma vienodai i=2:N-1. Kai i=1 ir i=N, eilutėse yra kraštinės sąlygos, tad čia bus vienetinės matricos. Toliau pildome $\frac{\partial^2 u}{\partial y^2}$ aproksimacijas; jos yra išilgai y ašies, kas N*N matricoje reiškia ėjimą žingsniu N, pildant mazgų aproksimacijas gretimais išilgai y mazgais. Pildymo schema pavaizduota 2.4. paveiksle.



2.4 pav. 2D uždavinio matricos sudarymas bendru atveju

Vienmačio nestacionaraus (pradinio-kraštinio) uždavinio atveju norint gauti $O(k^p)$ ir $O(h^{2m})$ eilių aproksimacijas nauojant *FTBC* metodo modifikacijas, tinklelio viduje, kur pakanka taškų kraštuose centrinei aproksimacijai, naudosime tokią (p+1)*(2m+1) gardelę:



2.5 pav. 1Dt uždavinio apibendrinta aproksimavimo schema

Kairėje ir dešinėje arti krašto, kur nepakanka taškų centrinei aproksimacijai, atitinkamai naudosime modifikuotą x atžvilgiu gardelę su dešininiais ir kairiniais skirtuminiais santykiais vietoje centrinių.

2.2. Ekstrapoliacija. Koeficientų radimas

Čia iškyla problema, kaip gauti pradines sąlygas laiko aproksimacijai, t.y. K+1:K+p-1 sluoksnius, kai p>1. Jei skaičiuotume jas mažesniu tikslumu O(k), tai sumažėtų ir galutinio sprendinio tikslumas,nebegautume norimos eilės t atžvilgiu aproksimacijos. Todėl reikia gauti pradines sąlygas algoritmui $O(k^p)$ tikslumu. Tam pasinaudojame ekstrapoliacijos idėja.

K+1:K+p-1 sluoksnius reikia aproksimuoti $O(k^p)$ tikslumu, turint K-tąjį sluoksnį $O(k^p)$ tikslumu (K=0 sluoksnis – pradinė sąlyga, duotas tiksliai).

Tegu U_j^t žymi t žingsniu (erdvės kintamojo žingsnis h fiksuotas) skaičiuotą sprendinio aproksimaciją j-tajame sluoksnyje, o u_j - tikroji sprendinio reikšmė t žingsnio tinklelio tame pačiame sluoksnyje. Erdvės kintamojo pozicijos neišskiriame, nes paklaidų eilė atžvilgiu t nepriklauso nuo erdvės kintamojo mazgo pozicijos. Sakykime, apskaičiuojame aproksimacijas vis mažesniu laiko žingsniu (mažesnio laiko žingsnio tinklelyje), tuomet pradiniame tinklelyje (kai laiko žingsnis t) teisingos lygybės:

$$U_j^t = u_j + C_1 t + C_2 t^2 + \cdots \mid * a_1;$$

$$U_{2j}^{t/2} = u_j + C_1 \frac{t}{2} + C_2 \left(\frac{t}{2}\right)^2 + \cdots \mid * a_2; \qquad (2.2)$$

…

$$U_{2^{p-1}j}^{t/2^{p-1}} = u_j + C_1 \frac{t}{2^{p-1}} + C_2 \left(\frac{t}{2^{p-1}}\right)^2 + \cdots \mid * a_p;$$

Reikia su tam tikrais svoriais sudėti gautas lygybes taip, kad pasinaikintų paklaidos prie žemiausių eilių t laipsnių. Naudojame neapibrėžtųjų koeficientų metodą. Sprendinio įvertis:

$$\widehat{U_j} = (a_1 + a_2 + \cdots + a_p)u_j + \left(1a_1 + \frac{1}{2}a_2 + \cdots + \frac{1}{2^{p-1}}a_p\right) tC_1 + \left(1a_1 + \left(\frac{1}{2}\right)^2 a_2 + \cdots +$$

$12p-12apt2C2+...=uj+0tC1+0t2C2+...$ (2.3)

Dešiniojoje lygybės pusėje - siekiamas rezultatas. Kad sistema turėtų vienintelį sprendinį, apribojame lygčių skaičių iki tiek, kiek yra nežinomųjų:

$$(a_1 + a_2 + \cdots + a_p)u_j + \left(1a_1 + \frac{1}{2}a_2 + \cdots + \frac{1}{2^{p-1}}a_p\right) tC_1 +$$

$$\left(1a_1 + \left(\frac{1}{2}\right)^2 a_2 + \cdots + \left(\frac{1}{2^{p-1}}\right)^2 a_p\right) t^2C_2 + \cdots + \left(1a_1 + \left(\frac{1}{2}\right)^{p-1} a_2 + \cdots + \left(\frac{1}{2^{p-1}}\right)^{p-1} a_p\right) t^{p-1}C_{p-1} = u_j +$$

$$0tC_1 + 0t^2C_2 + \cdots + 0t^{p-1}C_{p-1} \qquad (2.4)$$

Tuomet ekstrapoliuoto sprendinio t žingsnio tinklelyje paklaidų dedamosios prasidės nuo nario $C_p t^p$, taigi sprendinio įvertis $\widehat{U_j}$ bus $O(t^p)$ eilės, ir tokiam tikslumui gauti sprendinio aproksimacija bus skaičiuota p kartų tikslumu $O(t)$, - žingsniais $t, \frac{t}{2}, \frac{t}{4}, \dots, \frac{t}{2^{p-1}}$.

Ekstapoliacijos koeficientams rasti gauname tiesinių lygčių sistemą, kuri matricinėje formoje yra tokia:

$$\begin{pmatrix} 1 & 1 & 1 & & 1 \\ 1 & \frac{1}{2} & \frac{1}{2^2} & \dots & \frac{1}{2^{p-1}} \\ 1 & \left(\frac{1}{2}\right)^2 & \left(\frac{1}{2^2}\right)^2 & & \left(\frac{1}{2^{p-1}}\right)^2 \\ & \vdots & & \ddots & \vdots \\ 1 & \left(\frac{1}{2}\right)^{p-1} & \left(\frac{1}{2^2}\right)^{p-1} & \dots & \left(\frac{1}{2^{p-1}}\right)^{p-1} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.5}$$

Tuomet

$$\widehat{U_j} = a_1 U_j^t + a_2 U_{2j}^{t/2} + .. + a_p U_{2^{p-1}j}^{t/2^{p-1}}, \text{ čia } \widehat{U_j} = u_j + O(t^p). \tag{2.6}$$

Ekstrapoliaciją taip pat įgyvendinsime ir vienmačiam stacionariam uždaviniui, tuomet galėsime palyginti sprendimų sudėtingumą bei rezultatus, gautus ekstrapoliuojant iki tos pačios eilės tikslumo ir gautus naudojant skirtumines schemas su aukštesnių eilių skirtuminiais santykiais. Tam apskaičiuojamos sprendinių aproksimacijos h žingsnio tinklelyje tikslumu $O(h^2)$ – naudojamas mažiausio tikslumo metodas su centriniais skirtuminiais santykiais. Mažinant žingsnį gaunamos sprendinių aproksimacijos:

$$U_j^h = u_j + D_2 h^2 + D_4 h^4 \dots | * b_1;$$

$$U_{2j}^{h/2} = u_j + D_2 \left(\frac{h}{2}\right)^2 + D_4 \left(\frac{h}{2}\right)^4 \dots | * b_2; \tag{2.7}$$

…

$$U_{2^{m-1}j}^{h/2^{m-1}} = u_j + D_2 \left(\frac{h}{2^{m-1}}\right)^2 + D_4 \left(\frac{h}{2^{m-1}}\right)^4 \dots | * b_m;$$

Vėl panašiai, ekstapoliacijos koeficientai randami iš sistemos:

$$\begin{pmatrix} 1 & 1 & 1 & & 1 \\ 1 & \left(\frac{1}{2}\right)^2 & \left(\frac{1}{2^2}\right)^2 & \dots & \left(\frac{1}{2^{m-1}}\right)^2 \\ 1 & \left(\frac{1}{2}\right)^4 & \left(\frac{1}{2^2}\right)^4 & & \left(\frac{1}{2^{m-1}}\right)^4 \\ & \vdots & & \ddots & \vdots \\ 1 & \left(\frac{1}{2}\right)^{2(m-1)} & \left(\frac{1}{2^2}\right)^{2(m-1)} & \dots & \left(\frac{1}{2^{m-1}}\right)^{2(m-1)} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_m \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.8}$$

Tuomet

$$\widehat{U_j} = b_1 U_j^h + b_2 U_{2j}^{h/2} + .. + b_m U_{2^{m-1}j}^{h/2^{m-1}}, \text{ čia } \widehat{U_j} = u_j + O(h^{2m}). \tag{2.9}$$

Taigi tokiam tikslumui gauti sprendinio aproksimacija skaičiuojama m kartų tikslumu $O(h^2)$, - žingsniais $h, \frac{h}{2}, \frac{h}{4}, \dots, \frac{h}{2^{m-1}}$.

2.3. Rezultatai

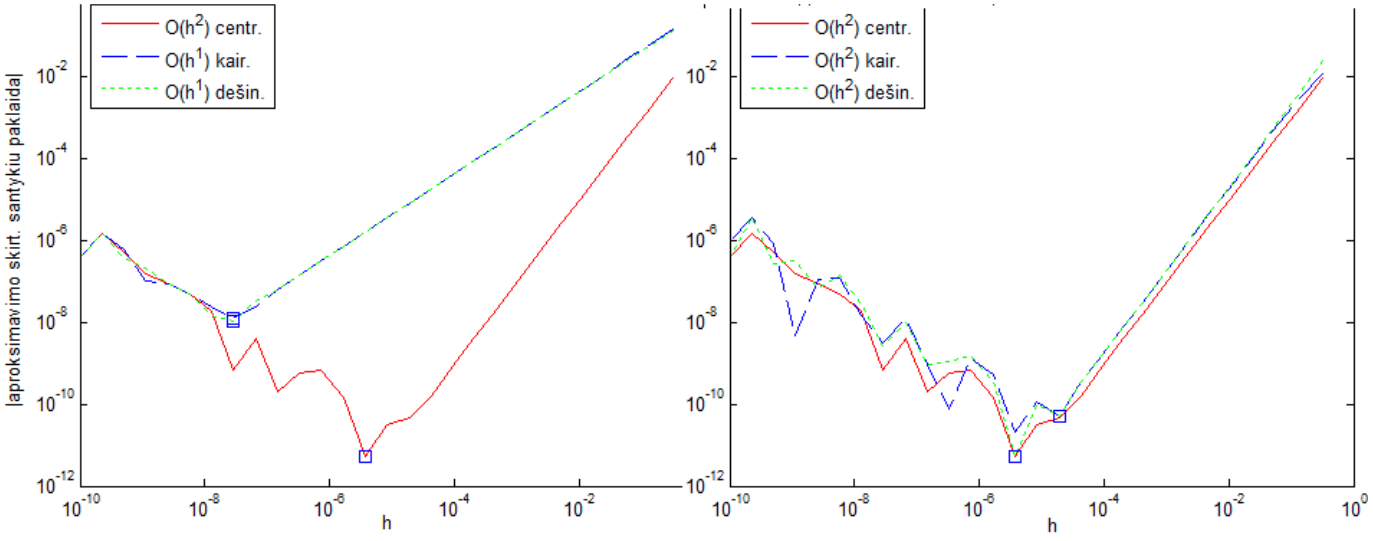
2.3.1. Išvestinių pakeitimo skirtuminiais santykiais paklaidos

Pirmiausiai panagrinėsime ne diferencialinių lygčių aproksimavimo paklaidas, bet pavienių išvestinių. Imame konkrečią funkciją, jos išvestinė žinoma; fiksuojame tašką x_0, kuriame tirsime aproksimacijos tikslumą. Išvestinę tame taške pakeičiame tam tikru pasirinktu skirtuminiu santykiu (metodą parašėme bendram atvejui). Keisdami tinklelio žingsnį h, galime stebėti, kaip keičiasi išvestinės pakeitimo skirtuminiu santykiu paklaida – paklaidą apskaičiuojame kaip absoliutinę reikšmę žinomos išvestinės reikmės taške x_0 ir aproksimacijos skaitinės reikšmės skirtumo. Teoriškai, jei skirtuminis santykis yra $O(h^p)$ eilės, tai paklaida turėtų mažėti proporcingai h^p.

Šiame tyrime rezultatai kokybine prasme nesiskiria kelioms nagrinėtoms skirtingoms funkcijoms (nes nagrinėjama sritis labai maža), todėl pateiksime rezultatus su viena funkcija:

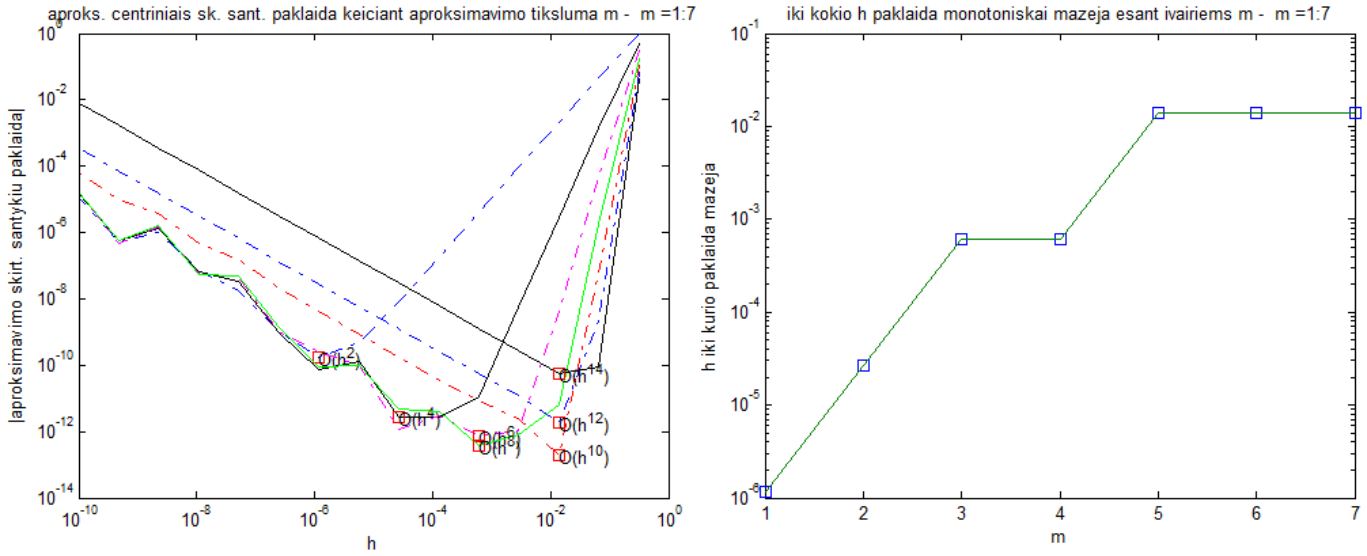
$$f = \cos(x),\ x_0 = 10$$

Grafikuose 1 parodyta paklaida $f'(x_0)$ pakeitus centriniais, kairiniais ir dešininiais kelių tikslumo eilių skirtuminiais santykiais. Skalė loglogaritminė, todėl nuolydžio tiesių koeficientas rodo kokiu greičiu paklaida kinta kintant žingsniui h.



2.6 pav. Išvestinės aproksimavimo skirtuminiais santykiais paklaida keičiant žingsnį h, loglogaritminėje skalėje.

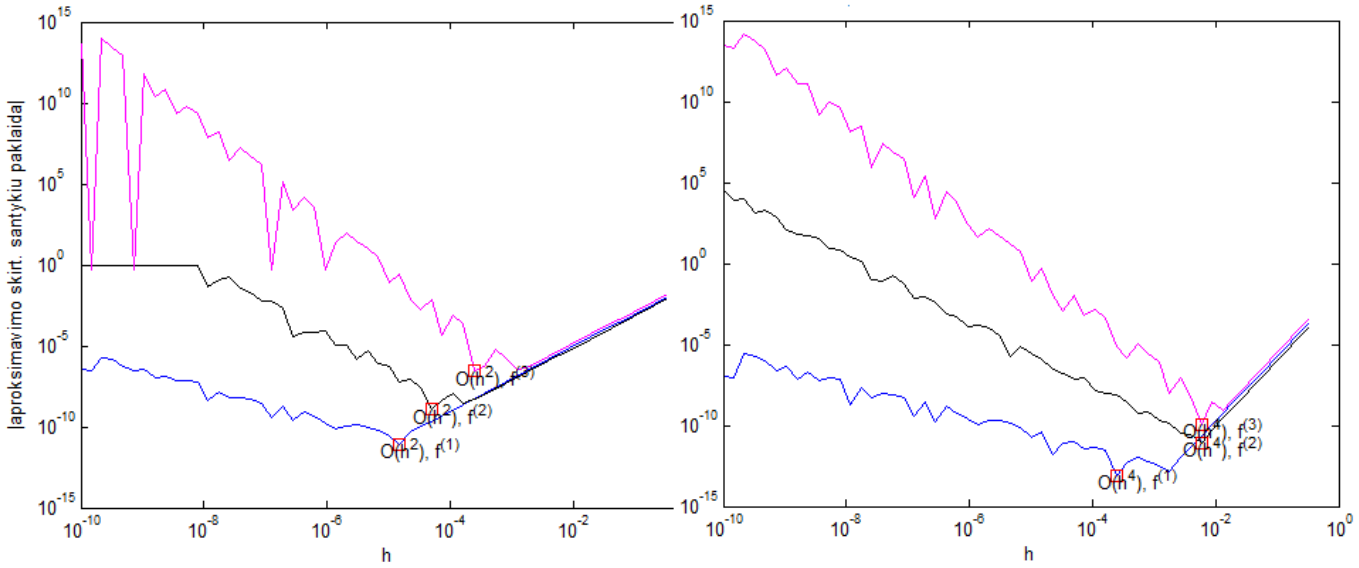
Grafike 2.1 parodyta, kaip kinta minėta paklaida, aproksimuojant centriniais įvairių eilių (konkrečiau- $O(h^2), O(h^4), \dots, O(h^{14})$) skirtuminiais santykiais; grafike 2.2 pavaizduota minimalaus žingsnio (didžiausio tinkelio smulkumo) priklausomybė nuo tikslumo parametro m, kur tikslumas $O(h^{2m})$.



2.7 pav. 1. Išvestinės aproksimavimo įvairių tikslumų centriniais skirtuminiais santykiais paklaidos keičiant žingsnį h, loglogaritminėje skalėje. 2. iki kokio h paklaida monotoniškai mažėja esant įvairiems m.

Minimalus žingsnis pasirinktas apskaičiuoti taip: tai žingsnis, iki kurio 2.1 grafike pavaizduota aproksimavimo paklaida monotoniškai mažėja, t.y. iki kol paklaidos kitimo skaitiškai apskaičiuota išvestinė pirmą kartą pakeičia ženklą. 2.2 grafike matome, kad didinant tikslumo parametrą m, turime rinktis vis didesnius žingsnius, t.y. stambesnį tinklelį, nes nuo tam tikro žingsnio (kurį ir rodo 2.2. grafikas) smulkindami tinklelį galime gauti blogesnius rezultatus.

2.6 ir 2.7 grafike pavaizduotos $f'(x_0)$ aproksimavimo paklaidos, o grafike 2.8 pavaizduota, kaip skiriasi $f'(x_0)$, $f^{(2)}(x_0)$ ir $f^{(3)}(x_0)$ pakeitimo centriniais skirtuminiais santykiais paklaidos. Matome, kad kaip ir pirmosios išvestinės aproksimavime, didinant tikslumą, mažėja žingsnis, iki kurio verta smulkinti tinklelį, siekiant mažesnių paklaidų. Taip pat matome, kad kuo aukštesnė išvestinės eilė, tuo minimalus žingsnis, iki kurio paklaidos monotoniškai mažėja, yra didesnis.



2.8 pav. Išvestinių aproksimavimo įvairių tikslumų centriniais skirtuminiais santykiais paklaidos keičiant žingsnį h, loglogaritminėje skalėje.

Rezultatus galima paaiškinti taip. Keičiant išvestinę skirtuminiu santykiu, atsiranda dviejų rūšių paklaidos. Jau minėta lokalioji paklaida, atsirandanti dėl begalinės eilutės pakeitimo baigtine suma, pavyzdžiui tikslią funkcijos išvestinės taške x_0 išraišką $f'(x_0) = \frac{y_1 - y_0}{h} - \frac{f^{(2)}(\xi)}{2} h$ pakeitus apytikslia išraiška

$$f'(x_0) \approx \frac{y_1 - y_0}{h} \tag{2.10}$$

gaunama paklaida

$$\frac{f^{(2)}(\xi)}{2} h, \tag{2.11}$$

t.y. $O(h)$ (jei $f^{(2)}(\xi)$ aprėžta) – mažinant h, paklaida proporcingai mažės. Tačiau kai h bus tiek mažas, kad $y_1 \approx y_0$, t.y. $f(x_1) \approx f(x_0)$, tai bet kokia maža paklaida eps (2.10) formulėje, atsirandanti dėl kompiuterio aritmetikos baigtinumo skaičiuojant $f(x_i)$, tampa žymi, nes ji padauginama iš $1/h$, taigi kuo h mažesnis, tuo didesnę įtaką turi ši paklaida eps. Be to, iš šių samprotavimų išplaukia, kad egzistuoja optimalus žingsnis h, su kuriuo šių dviejų tipų paklaidų suma yra minimali:

$$\frac{eps_1 - eps_2}{h} + \frac{f^{(2)}(\xi)}{2} h \to \min_h \tag{2.12}$$

Aukštesnių eilių išvestinėms tas taškas pasiekiamas greičiau, nes aukštesnių eilių išvestinių aproksimuojančių skirtuminių santykių vardiklyje yra aukštesnės eilės h laipsnis, todėl pirmoji paklaida išauga greičiau.

Kuo didesnis h laipsnis (kuo teoriškai tikslesnė aproksimacija), tuo greičiau pasiekiamas tas taškas. Antroji paklaidos dedamoji turi vis mažesnę įtaką esant didesniam h laipsniui, o pirmoji dedamoji išauga greičiau, nes aproksimuojama per daugiau taškų, susikaupia daugiau apvalinimo paklaidų.

2.3.2. 1D stacionarus atvejis

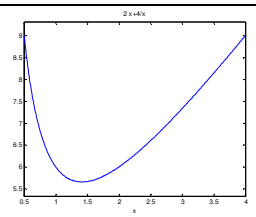
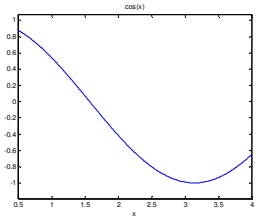
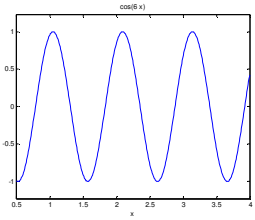
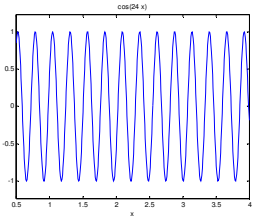
Sprendžiami 4 konkretūs uždaviniai su žinomais sprendiniais:

1: $a_1(x) = 1;\ a_2(x) = \frac{1}{x};\ a_3(x) = -\frac{1}{x^2};\ a_4(x) = 0;\ u(0.5) = 9; u(4) = 9$

2, 3, 4: $a_1(x) = -e^x;\ a_2(x) = -e^x;\ a_3(x) = x^3;\ a_4(x) = -x^3\cos(bx) - e^x\big(b\sin(bx) + b^2\cos(bx)\big);$

$x_0 = 0.5;\ x_f = 4$, sprendinys $u(x) = \cos(bx);\ b \in \{1, 6, 24\}$.

2.1 lentelė. 1D uždavinio sprendinių grafikai

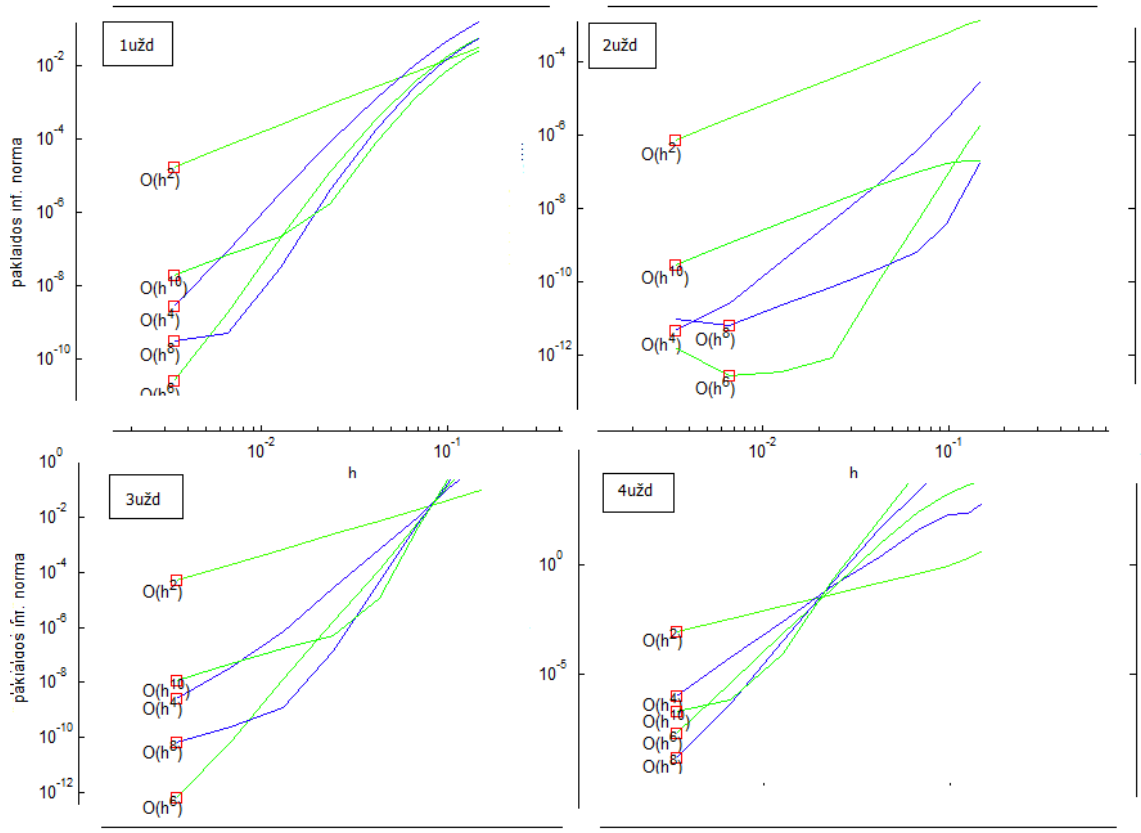
1. $u(x) = \frac{2x+4}{x}$	2. $u(x) = \cos(x)$	3. $u(x) = \cos(6x)$	4. $u(x) = \cos(24x)$
			

Gauname paklaidos priklausomybės grafikus kiekvienam uždaviniui (paklaida skaičiuojama kaip maksimumo norma) keičiant tiek schemos tikslumą, tiek žingsnį – paklaidos pavaizduotos 2.9 paveiksle. Pirmų trijų uždavinių paklaidos mažesnės už 1, o 4-ojo uždavinio paklaidos, esant didesniems h, siekia 10^5.

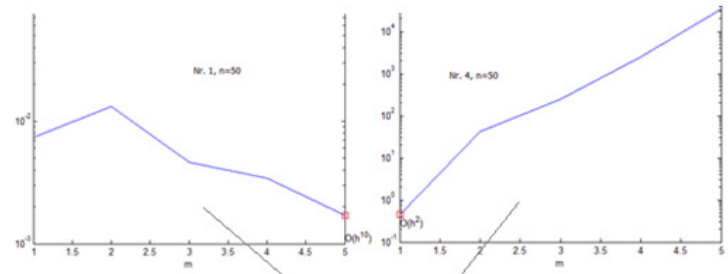
Randame eilę p=2m, su kuria paklaida minimali kiekvieno uždavinio atveju, t.y. optimalią tikslumo eilę. Rezultatai pavaizduoti 2.10 paveiksle.

Metodų stabilumą patikriname kiekvienai nagrinėjamai schemai. Kaip aprašyta teorinėje dalyje, stabilumą maksimumo normoj tiesiniam uždaviniui galima patikrinti skaičiuojant atvirkštinių matricų maksimumo normų seką, mažinant h, ir ji turi būti aprėžta. Visos nagrinėjamų uždavinių skirtuminės schemos pagal šį kriterijų yra stabilios – mažėjant h seka nusistovi prie 1, pvz. 4 uždaviniui gauti grafikai pateikti 2.11 paveiksle.

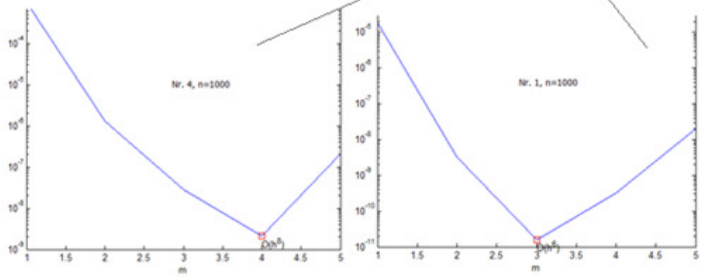
Vadinasi, remdamiesi teiginiu (1.34) teoriškai galima tikėtis, kad esant pakankamai mažiems h globalioji paklaida bus tos pačios eilės, kokios yra sudaryta skirtuminė schema.



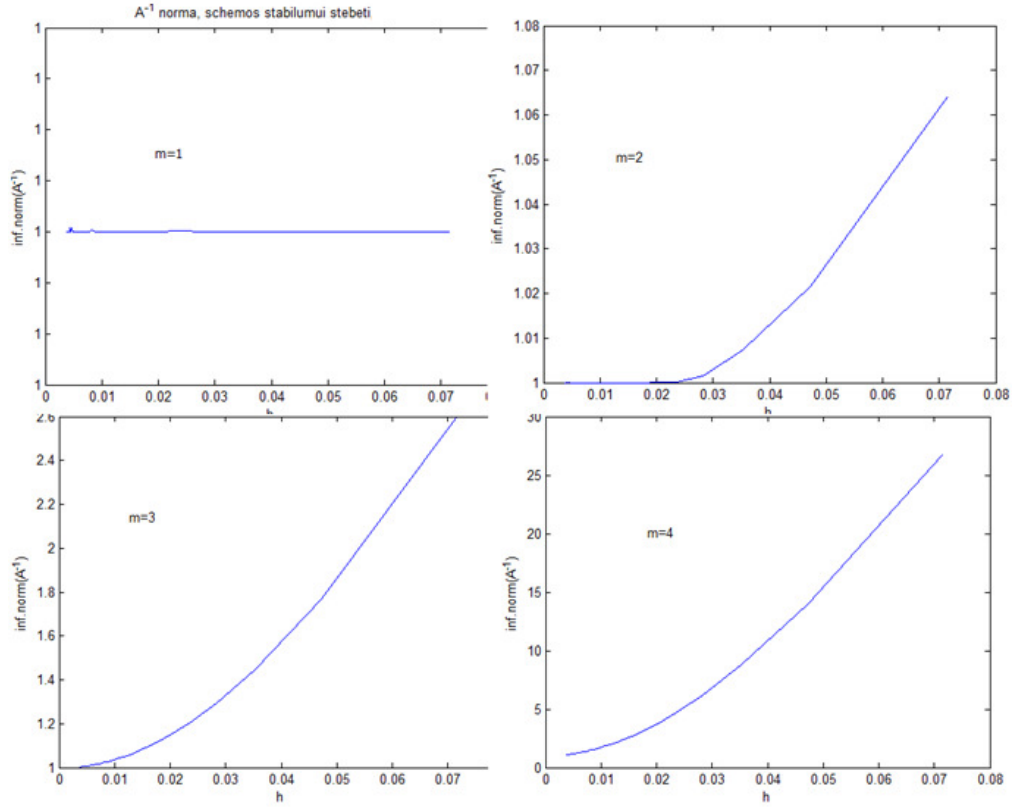
2.9 pav. Sprendimo paklaidų priklausomybės grafikai kiekvienam uždaviniui keičiant schemos tikslumo parametrą m ir tinklelio žingsnį h, loglogaritminėje skalėje.



Užd. \ n	22	50	100	500	1000	2000
1	10	10	10	8	6	4
2	10	8	6	6	6	6
3	2	10	10	6	6	6
4	2	2	2	10	8	6



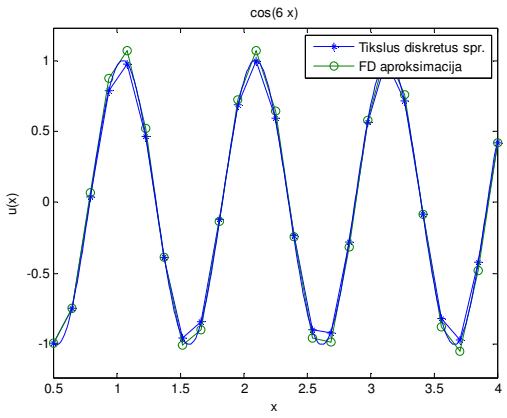
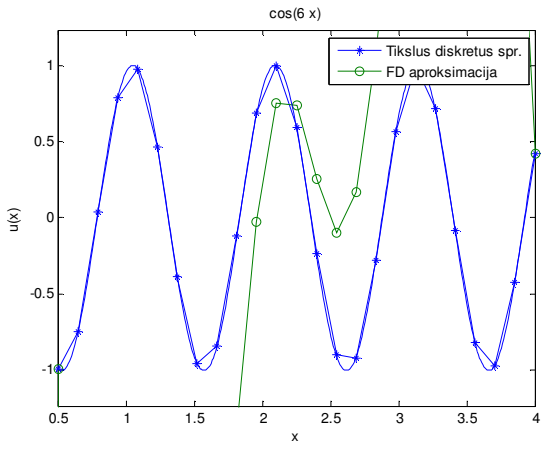
2.10 pav. Tikslumo eilė p=2m, su kuria paklaida minimali kiekvieno uždavinio atveju.

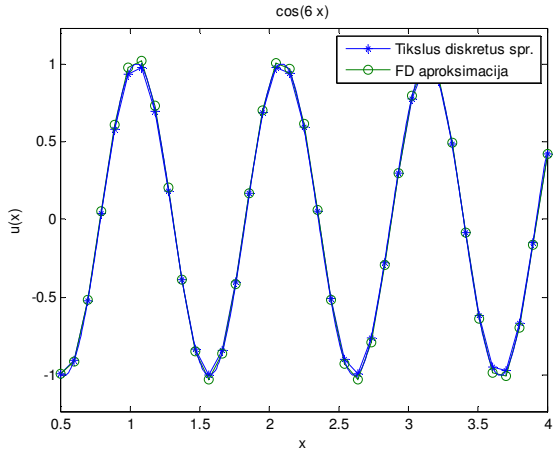
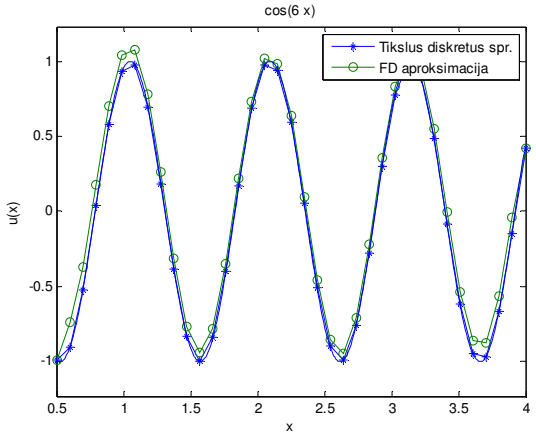
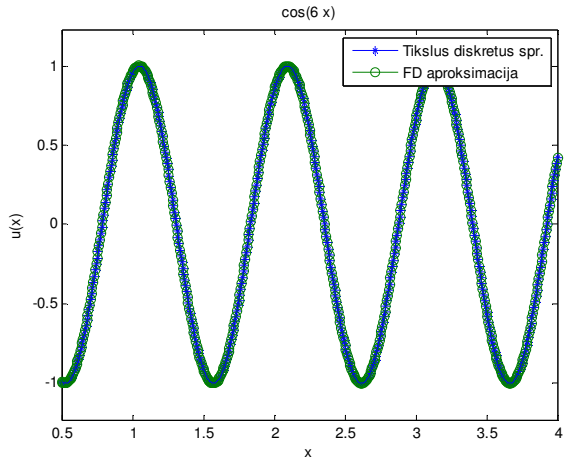
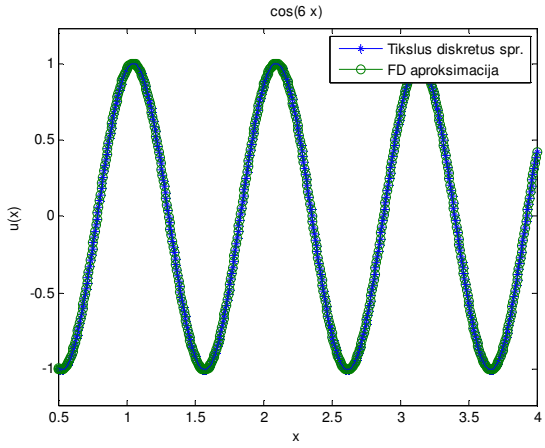
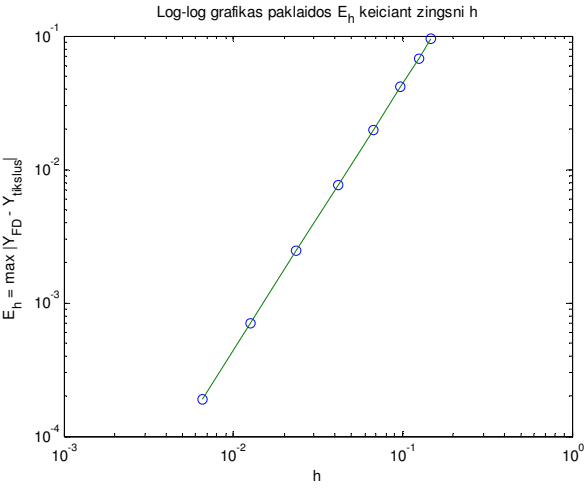
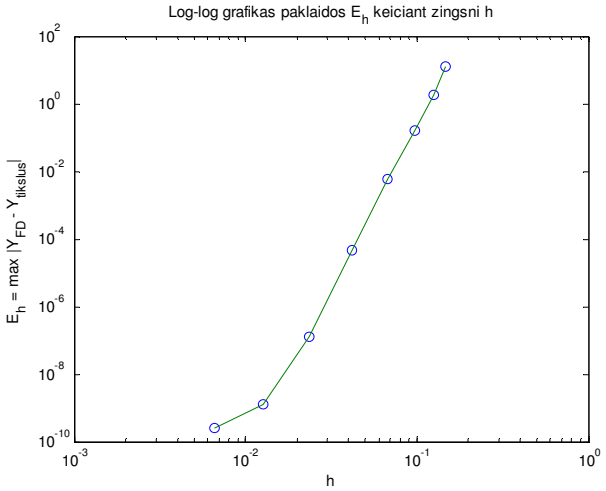


2.11 pav. 1D schemų skaitinis stabilumo tyrimas

Panagrinėkime 3 uždavinį. Apskaičiuosime paklaidas esant įvairiems žingsniams h ir įvertinsime konvergavimo greitį. Palyginimui rezultatau pateikti 2.2. lentelėje. Pirmame stulpelyje nurodytas n – intervalo dalinimų skaičius. Antrame ir trečiame stulpelyje pateikti sprendinių aproksimavimo grafikai atitinkamai naudojant mažiausio tikslumo schemą su m=1 ir didelio tikslumo schemą su m=4.

2.2 lentelė. 1D uždavinio nr.3. sprendinio aproksimavimo ir konvergavimo analizė esant kelioms tikslumo parametro m reikšmėms.

n	Užd3, m=1	Užd3, m=4
25		

37		
53 3		
		
	Konv. įvertis p = 2.028	Konv. įvertis p = 8.768e

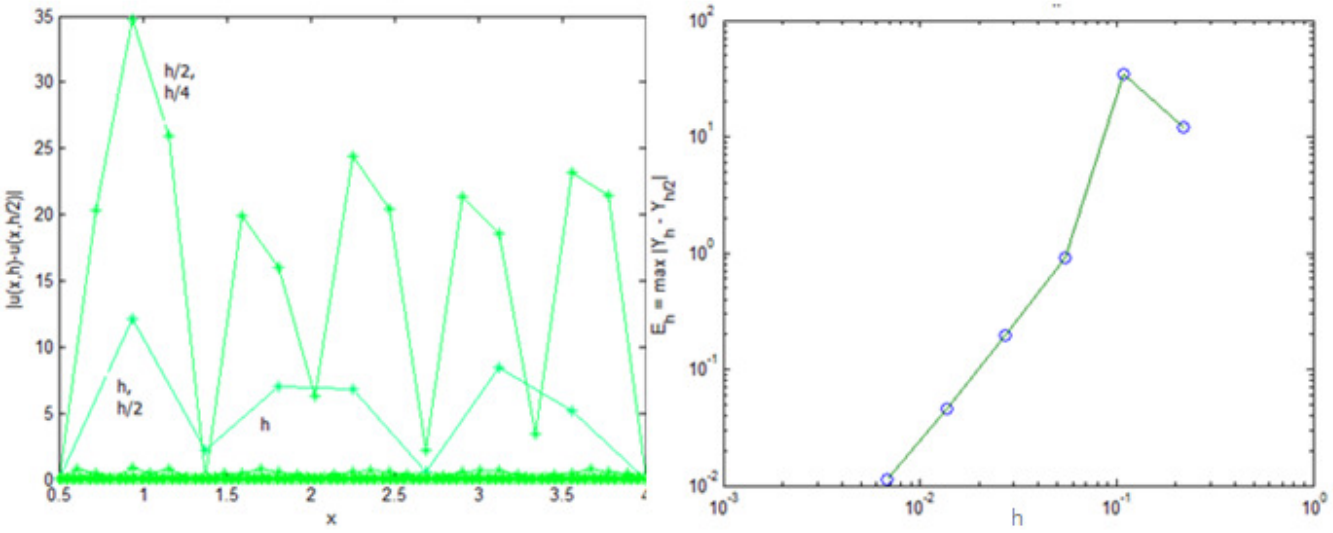
Matome, kad esant per dideliam žingsniui, lyginant su sprendinio kitimu, mažesnio tikslumo schema aproksimuoja tiksliau. Tą galima paaiškinti taip - aproksimacijoje imami tik artimiausi taškai, todėl

mažiau nuklystama. 2.9 pav. grafike matėme, kad 3-ajam ir ypač 4-ajam uždaviniui didesnio tikslumo schemos esant pakankamai dideliems žingsniams davė didesnes paklaidas nei mažesnio tikslumo schemos. Todėl rinktis didesnio tikslumo schemą ir mažesnį žingsnį nėra saugus būdas spręsti uždaviniui, kurio sprendinys nežinomas.

Sakykime, tikrojo sprendinio nežinome. Smulkindami tinklelį, galime palyginti sprendinius, gautus stambesniame ir smulkesniame tinkleliuose. Jei žinome, kad skirtuminė schema stabili, tai mažinant h, sprendinys artės prie tikrojo sprendinio. Vadinasi, įvertinus skirtumą tarp sprendinių, gautų žingsniu h ir h/2, jei skirtumas nebus artimas 0, galima manyti, jog žingsnis h yra per didelis sprendžiamam uždaviniui.

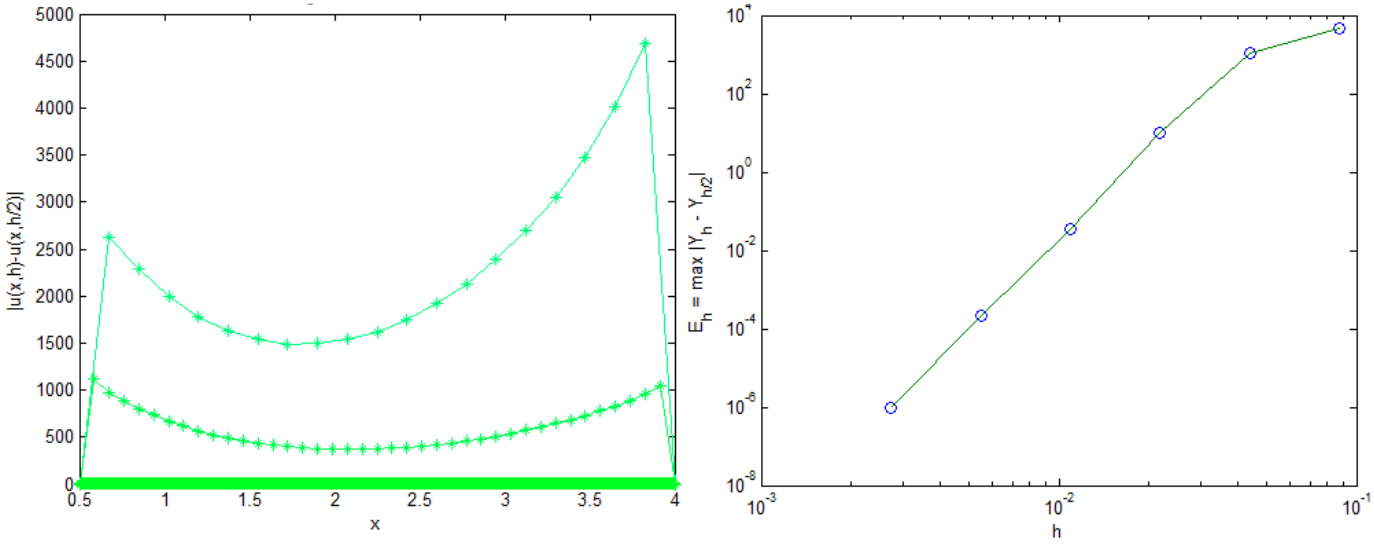
Be to, paklaidą apskaičiavę tokiu būdu $P(h) = u^h - u^{h/2}$, galime įvertinti konvergavimo greitį, neturėdami tikrojo sprendinio: $P(h) \approx Ch^p - C\left(\frac{h}{2}\right)^p$, $P\left(\frac{h}{2}\right) \approx C\left(\frac{h}{2}\right)^p - C\left(\frac{h}{4}\right)^p$, taigi $\frac{P(h)}{P\left(h/2\right)} \approx 2^p$, ką gauname ir lygindami su tiksliu sprendiniu.

Pvz, 4-ajam uždaviniui, imdami m=1, gauname gretimais žingsniais apskaičiuotų sprendinių skirtumų modulius. Tik trečioje iteracijoje, kai skaičiuota žingsniais $h_0/4$ ir $h_0/8$, sprendinių skirtumo modulis tapo artimas nuliui, taigi tinklelis tapo pakankamai smulkus nagrinėjamam uždaviniui – kurio sprendinys turi didelį dažnį. Šalia pavaizduotame konvergavimo grafike matome, kad, neskaitant pirmojo taško, konvergavimo greitis, kaip ir teoriškai galima tikėtis su tokio tikslumo schema, apytiksliai lygus 2. Tik reikėtų pastebėti, kad paklaida, gauta lyginant gretimais žingsniais apskaičiuotus sprendinius, yra didelė, o tai dar vienas požymis, kad toks mažas dalijimų skaičius šiam uždaviniui yra netinkamas.



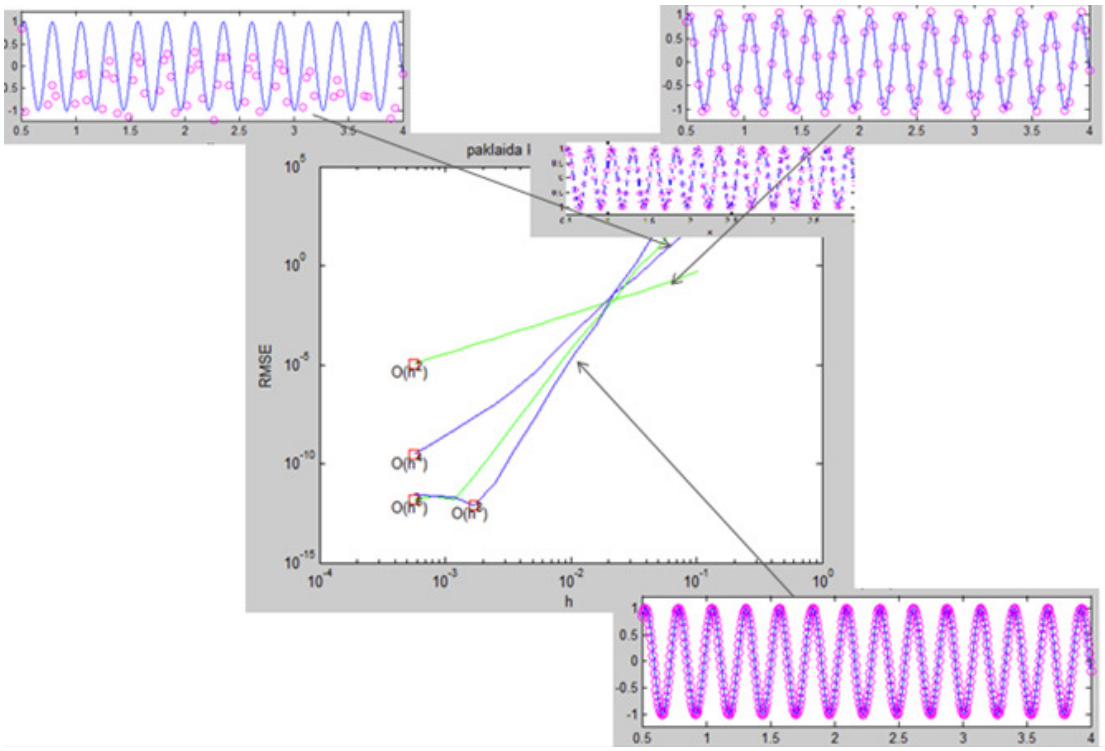
2.12 pav. Aproksimavimo ir konvergavimo tyrimas, kai sprendinys nežinomas, su parametru m=1

Be to, aukštesnių eilių schemos per stambiam tinkleliui yra dar jautresnės, pvz. tas pats uždavinys su m=3 – esant netinkamam žingsniui paklaidos žymiai didesnės nei su mažesnio tikslumo schema, t.y. m=1 (aproksimuota naudojant daugiau neteisingų taškų).



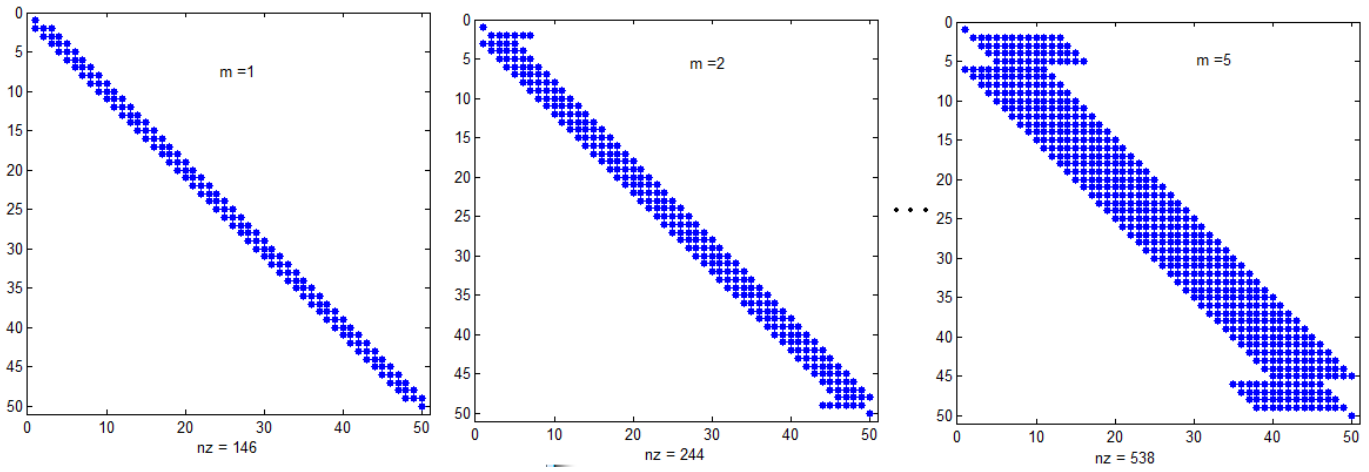
2.13 pav. Aproksimavimo ir konvergavimo tyrimas, kai sprendinys nežinomas, su parametru m=3

Dabar aišku, kodėl pirmajame grafikų rinkinyje 4 uždavinyje aukštesnių tikslumų schemos prie didelių žingsnių duoda didesnes paklaidas nei paprastesnės schemos:



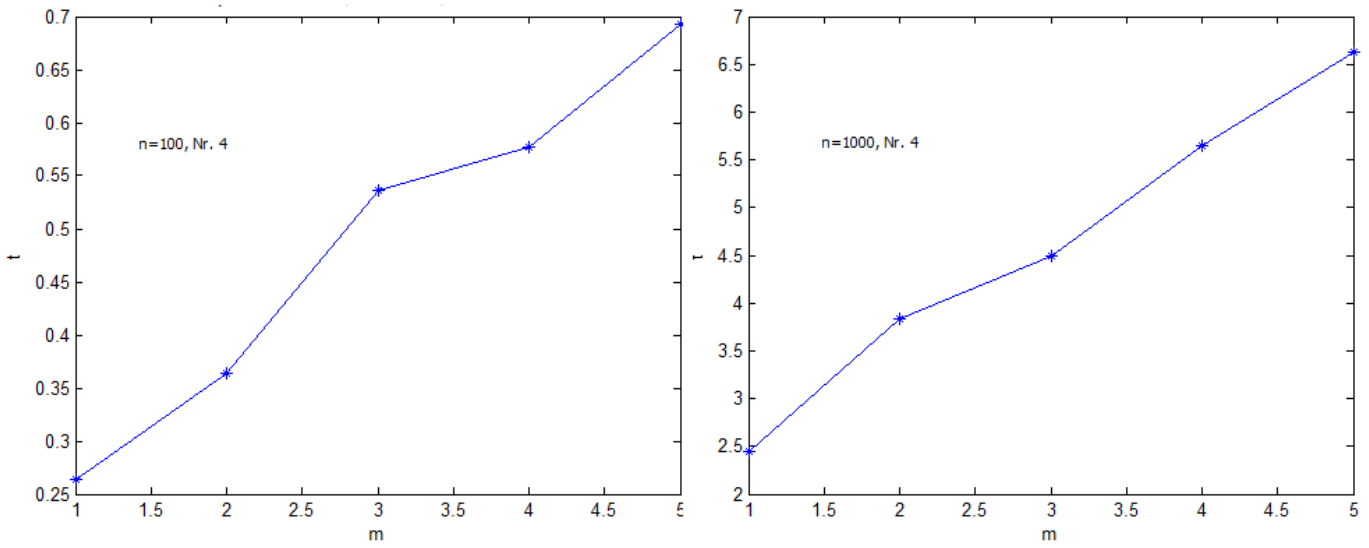
2.14 pav. 1D uždavinio nr.4 sprendimo paklaidų vizualinis tyrimas

Toliau parodyta, kaip sudėtingėja matrica, kai dalinimų skaičius yra n=50, o schemos tikslumo parametras m keičiamas nuo 1 iki 5; apačioje parašyta, kiek nenulinių elementų yra matricoje.



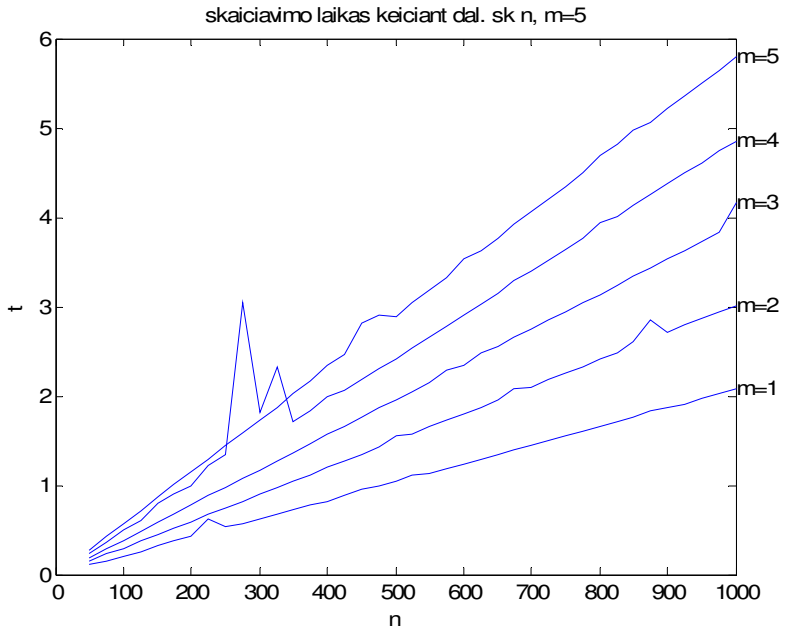
2.15 pav. 1D schemos matricos sudėtingumas esant įvairioms parametro m reikšmėms

Apžvelgsime skaičiavimų sudėtingumą laiko atžvilgiu.



2.16 pav. 1D uždavinio nr. 4 sprendimo laikas, kai dalinimų skaičius n=100, n=1000, m- keičiamas

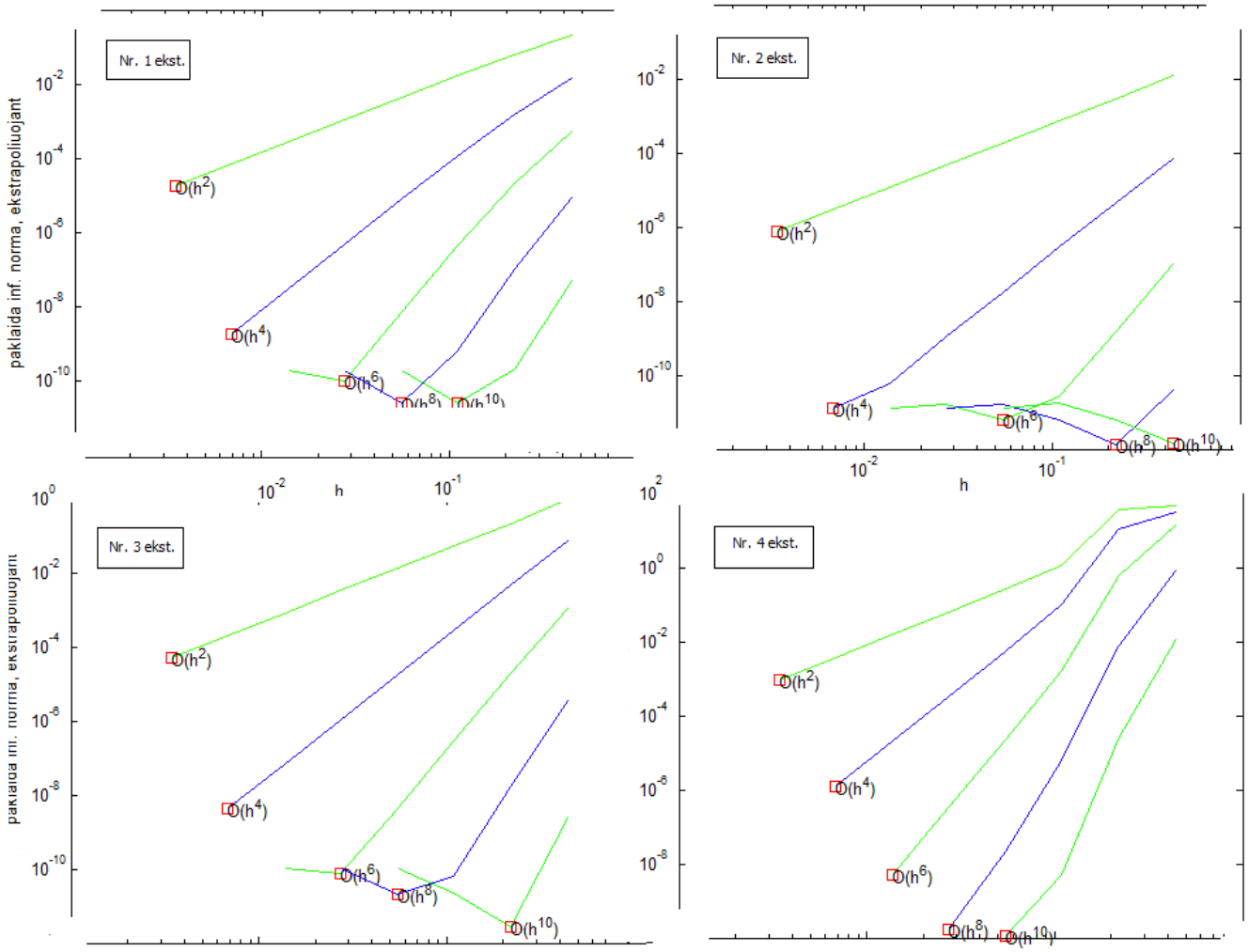
2.16 paveiksle galima pastebėti tendenciją, kad skaičiavimų laikas tiesiškai priklauso nuo schemos sudėtingumo parametro m, be to, priklauso nuo kompiuterio, ir net esant tam pačiam kompiuteriui, priklauso nuo jo procesoriaus užimtumo – pakartojus programą su tais pačiais parametrais, laiko grafikai šiek tiek skirsis, nors ir skaičiavimai buvo atlikti *safe mode* aplinkoje. Iš šių grafikų taip pat matome, kad n padidinus 10 kartų, laikas taip pat padidėjo apie 10 kartų. Randame laiko priklausomybę nuo dalinimų skaičiaus, esant parametro reikšmėms m=1, 2. ...,5:



2.17 pav. 1D uždavinio nr. 4 sprendimo laiko priklauomybė nuo dalinimų skaičiaus n ir tikslumo parametro m

Kadangi nagrinėjamam uždaviniui realizavome ir vadinamąją sprendinio įverčio ekstrapoliaciją, tai galime palyginti rezultatus, gautus įvairių tikslumų schemomis ir atitinkamo gylio ekstrapoliavimu.

Iškyla problema, kad norint nubraižyti analogiškus 2.9 pav. grafikus, to padaryti negalime, nes pritrūksta operatyvinės atminties - *Matlab* programa leidžia saugoti iki maždaug 10000*10000 dydžio double tipo kintamųjų matricą, o norint pasiekti p=2m tikslumą, reikia tą patį uždavinį spęsti žingsniais $h, \frac{h}{2}, \ldots \frac{h}{2^{m-1}}$, taigi pvz., jei nagrinėtume uždavinį ilgio L intervale, tai nepriklausomai nuo schemos tikslumo galėtume žingsnį mažinti iki $\frac{L}{10^4}$, o ekstrapoliuojant, kuo aukštesnės eilės tikslumo siekiame, tuo mažiau galime mažinti žingsnį. Jei siekiame $O(h^{2m})$ tikslumo, minimalus žingsnis bus apie $\frac{L}{2^{m-1}10^4}$, taigi apribojamos žingsnio mažinimo galimybės (tas ypač aktualu sprendžiant dvimatį uždavinį, nes jame matrica yra $N^2 * N^2$, taigi vieno kintamojo kryptimi galime atlikti tik iki maždaug 100 dalinimų, o norint ekstrapoliuoti $O(h^{2m})$ tikslumu, maksimalų dalinimų skaičių tektų dar mažinti 2^{m-1} kartų). Todėl 2.18 pav. pateiksime analogiškus grafikus, kai dalinimų skaičius yra pakoreguojamas pagal siekiamą tikslumą, taigi didesnio tikslumo sprendinių paklaidos pateiktos mažesniame h intervale.

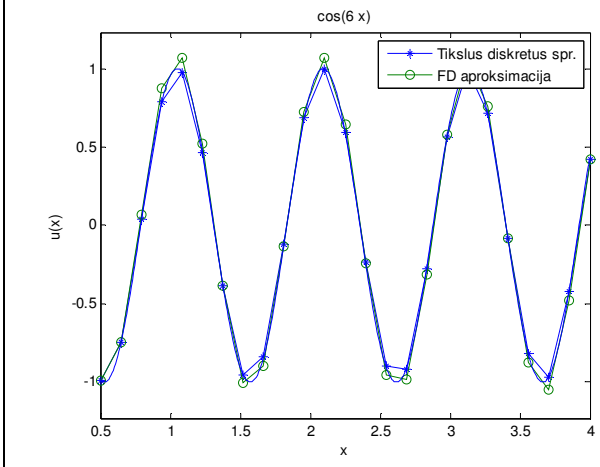
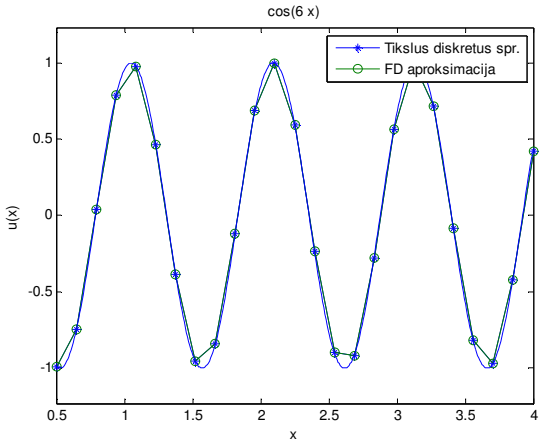
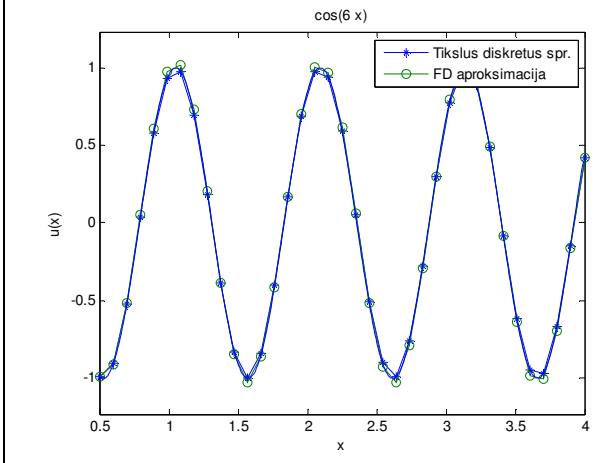
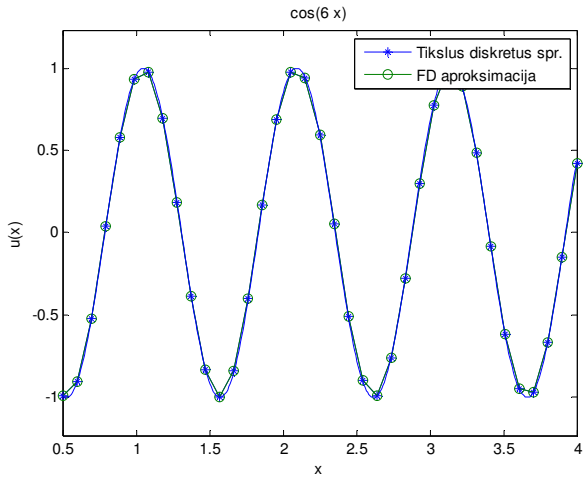
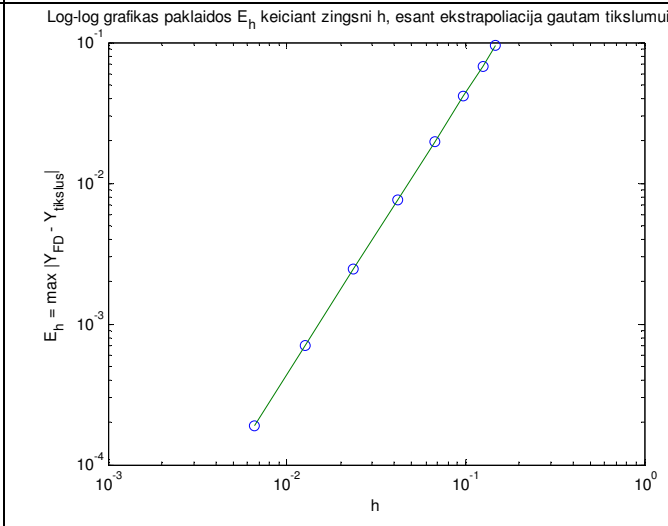
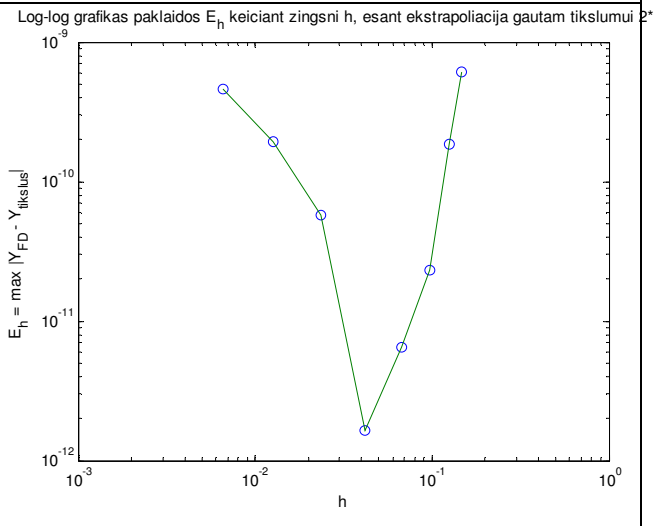


2.18 pav. Sprendimo paklaidų gautų naudojant ekstrapoliaciją priklausomybės grafikai kiekvienam uždaviniui keičiant m ir h, loglogaritminėje skalėje.

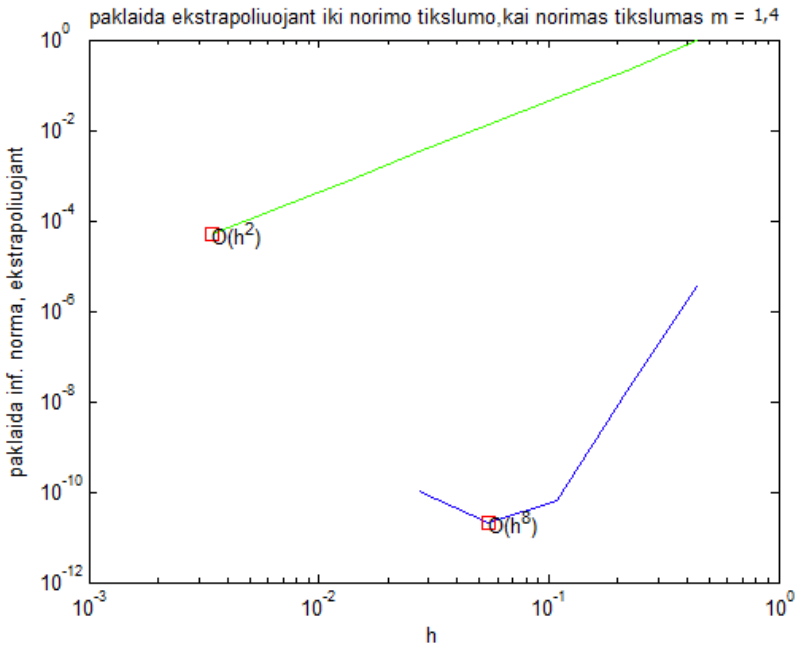
Palyginę matome ekstrapoliacijos metodo privalumą – aukštesnės eilės tikslumo metodas per dideliam pradiniam žngsniui duoda mažesnes paklaidas; pasirinkdami ekstrapoliacijos metodą didesniam tislumui gauti, galime būti tikresni, kad geresnį įvertį ir gausime, priešingai nei didesnių tikslumų skirtuminėse schemose. Aišku, čia galimas paprastas paaiškinimas – nors pradinis žingsnis mažesnis, tačiau tam tikslumui pasiekti schema skaičiuojama ir su mažesniais žingsniais, o tai ir ištaiso pirmųjų iteracijų – su netinkamais žingsniais – paklaidas. Dar verta pastebėti, kad jei teoriškai įrodytas ekstrapoliacijos metode naudojamos mažos tikslumo eilės schemos stabilumas, tai ir ekstrapoliuotas sprendinys bus stabilus, nes imame stabilių sprendinių sumą su tam tikrais svoriais.

Palyginame 3 uždavinio sprendinio aproksimavimą didesniu tikslumu naudojant didesnio tikslumo schemą bei nauojant ekstrapoliaciją, esant įvairiems intervalų dalinimų skaičiams. Tam palyginamos 2.2 ir 2.3 lentelės.

2.3 lentelė. 1D uždavinio nr.3. sprendinio, gauto ekstrapoliacijos metodu, aproksimavimo ir konvergavimo analizė esant kelioms tikslumo parametro m reikšmėms.

n	Užd3, m=1	Užd3, m=4
25		
37		
		
	p = 2.028	p = 5.575 mažėjimo intervale

Iš didesniu tikslumu ekstrapoliacijos būdu apskaičiuoto sprendinio konvergavimo grafiko matyti, kad paklaidos mažėjo tik iki tam tikro žingsnio, mažesniu greičiu nei turėtų teoriškai (4 eilės metodas turėtų konverguoti su p=8), o vėliau ima didėti. Tačiau paklaidos vis tiek yra mažesnės, nei skaičiuojant su mažiausio tikslumo schema, kas ypač aiškiai matosi 2.19 grafike, t.y. ekstrapoliacijos tikslas pasiektas.

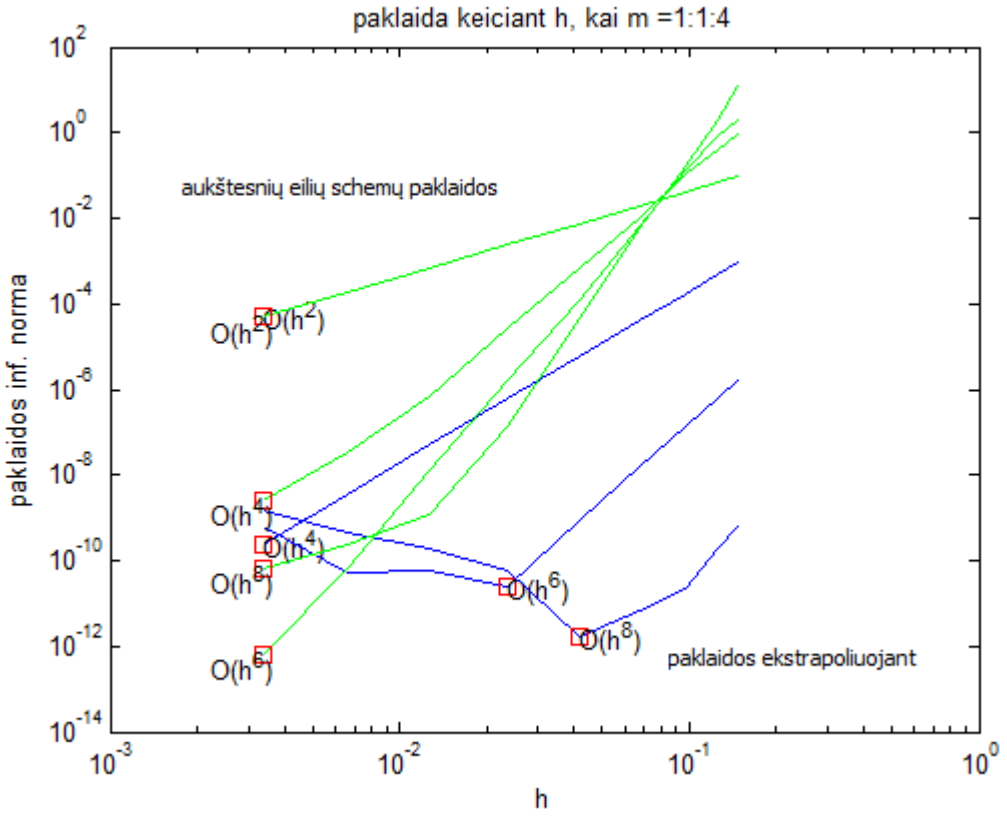


2.19 pav. Sprendimo paklaidos, gautos sprendinį ekstrapoliuojant iš p=2 tikslumo schemos iki tikslumo p=8, ir paklaidos, gautos su p=2 tikslumo schema

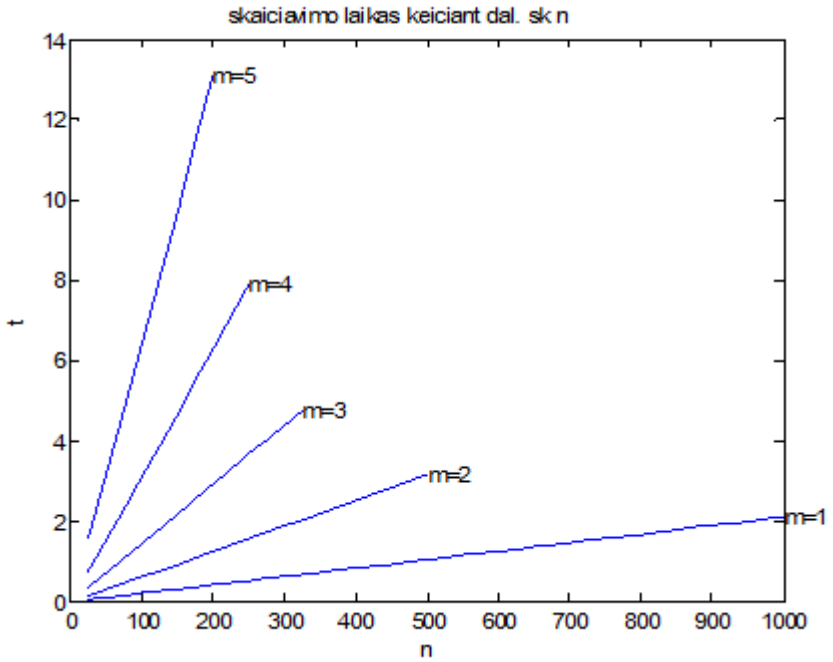
2.20 pav. palyginame paklaidas tą pačią tikslumo eilę bandant gauti ekstrapoliuojant bei naudojant aukštesnės eilės schemas; tik atkrepsime dėmesį, kad grafike atidėtos paklaidos ties pradiniu ekstrapoliacijos žingsniu, tad realiai tai paklaidai gauti buvo naudoti ir smulkesni tinkeliai. Iš šio grafiko matome, kad didelio tikslumo schemomis galima gauti keliom eilėm geresnius rezultatus, tinkamai parinkus optimalaus tikslumo schemą, tačiau skirtumas nėra didelis turint omeny tai, kad realiai yra nežinoma, kuri schema nagrinėjamam uždaviniui bus tiksliausia.

Priede 2.1 pateiktas ekstrapoliacijos būdu ir aukštesnių eilių schemomis gautų sprendinių konvergavimo greičių įverčių per 9 iteracijas žingsniu $2 * 2^k + 21$, k=1,..,9 palyginimas.

Įdomu palyginti ekstrapoliacijos metodo sudėtingumą laiko prasme. Kai m=1 uždaviniai sutampa, nes ekstrapoliacija nevykdoma. Kai ekstrapoliacija vykdoma, tai uždavinys sprendžiamas keletą kartų. Pvz. iš 2.17 schemų laiko grafiko matome, kad norint su n=500 gauti m=2 tikslumą ekstrapoliuojant, reikėtų spręsti m=1 uždavinį su žingsniais n=500 ir n=1000, ir šių laikų suma būtų apytikslė laiko trukmė ekstrapoliaciniam uždaviniui prie n=500, m=2, kas ir matyti 2.21 pav.



2.20 pav. Paklaidos, gautos aukštesnių eilių skirtuminėmis schemomis ir paklaidos, gautos ekstrapoliuojant iki norimo tikslumo.



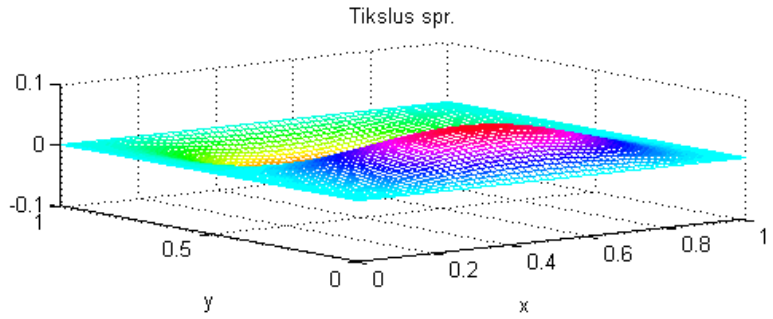
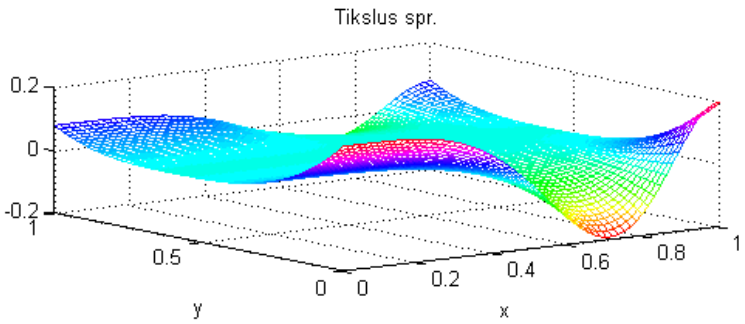
2.21 pav. 1D ekstrapoliacinio metodo sprendimo laiko priklauomybė nuo dalinimų skaičiaus n ir tikslumo parametro m

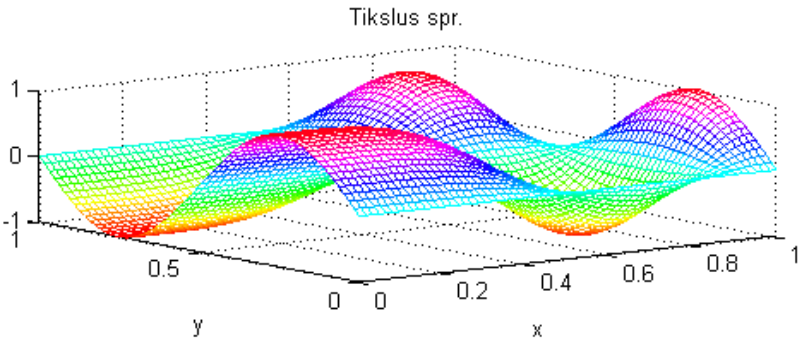
Taigi atsižvelgę į tai, kad su ekstrapoliacijos būdu gautų aproksimacijų paklaidos iš esmės nėra geresnės, o laiko sąnaudos didesnės, galime daryti prielaidą, kad vienmačiam stacionariam uždaviniui geriau rinktis aukštesnės tikslumo eilės schemas, ypač tuomet, kai sprendinio aproksimacijos reikia kuo smulkesniame tinklelyje (pvz., kad būtų galima vėliau interpoliuoti didesniu tikslumu). Tačiau kaip jau minėta, ekstrapoliacija pravers vienmačio nestacionaraus uždavinio inicializavimui.

2.3.3. 2D stacionarus atvejis

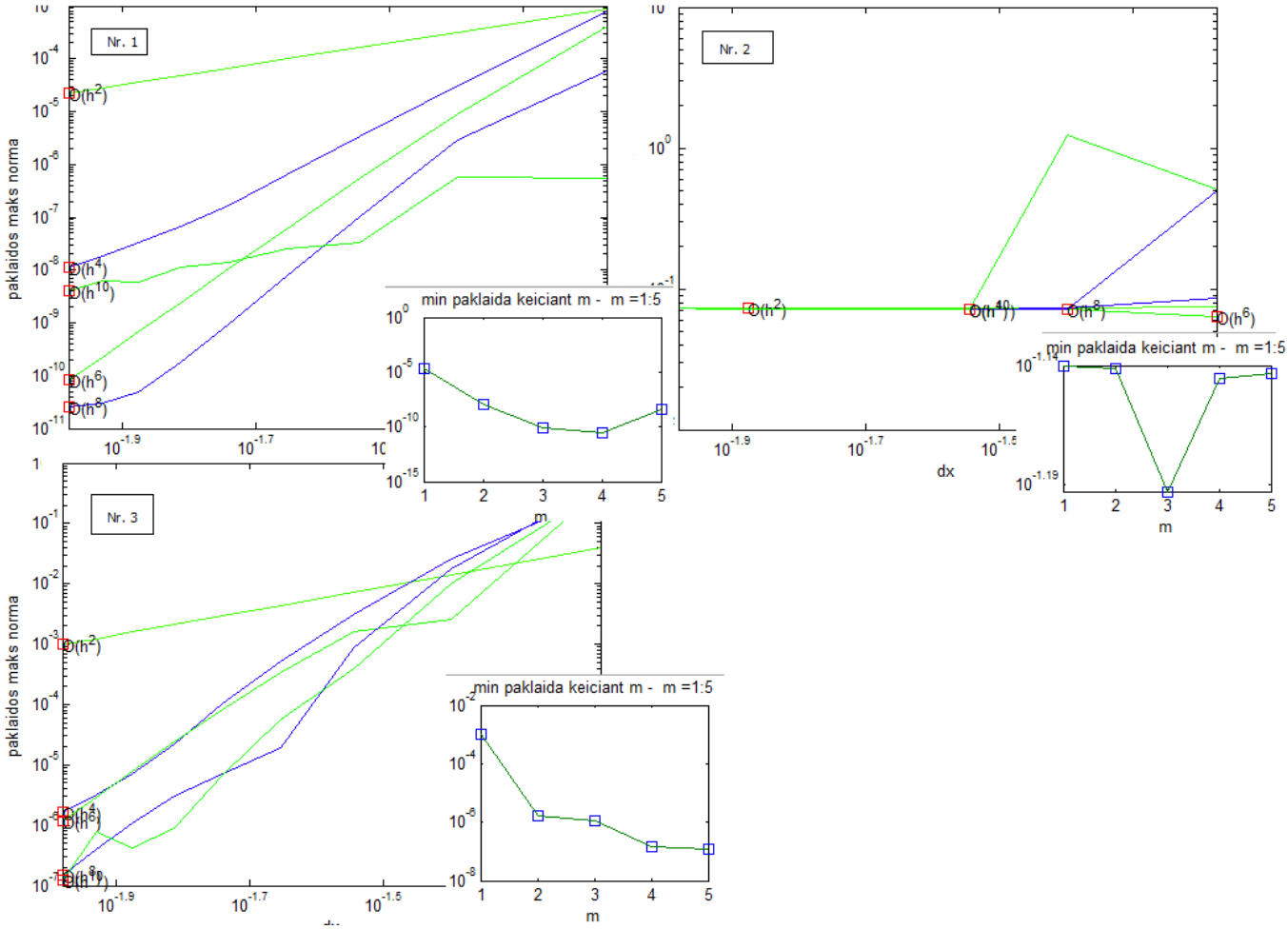
Sprendžiami 3 konkretūs Puasono kraštinio uždavinio (1.54) vienetiniame kvadrate Ω =[0, 1]*[0, 1] atvejai, kai tikslūs sprendiniai yra žinomi.

2.4 lentelė. Nagrinėjami 2D kraštiniai uždaviniai ir jų sprendiniai.

Nr.	uždavinys	sprendinys
1	$f(x,y) = 4\pi^2 \sin(2\pi x) y^2 (1 - y)^2 - \sin(2\pi x)(2 - 12y + 12y^2)$ $u(x,0) = 0, u(x,1) = 0,$ $u(0,y) = 0, u(1,y) = 0.$	$u(x,y) = \sin(2\pi x) y^2 (1-y)^2$ 
2	$f(x,y) = -6\lvert x - \sqrt{1/3}\rvert \cos(2\pi y^2) - \lvert x - \sqrt{1/2}\rvert^3 (-16\pi^2 y^2 \cos(2\pi y^2) - 4\pi \sin(2\pi y^2))$ $u(x,0) = \lvert x - \sqrt{1/3}\rvert^3,$ $u(x,1) = \lvert x - \sqrt{1/3}\rvert^3,$ $u(0,y) = (\sqrt{1/3})^3 \cos(2\pi y^2),$ $u(1,y) = \lvert 1 - \sqrt{1/3}\rvert^3 \cos(2\pi y^2)$	$u(x,y) = \lvert x - \sqrt{1/3}\rvert^3 \cos(2\pi y^2)$ 

3	$f(x,y) = 4\pi\sin(2\pi x)\left(\pi\cos(2\pi y^2)(1+4y^2)+\sin(2\pi y^2)\right)$ $u(x,0)=\sin(2\pi x)$, $u(x,1)=\sin(2\pi x)$, $u(0,y)=0, u(1,y)=0$.	$u(x,y)=\sin(2\pi x)\cos(2\pi y^2)$. 

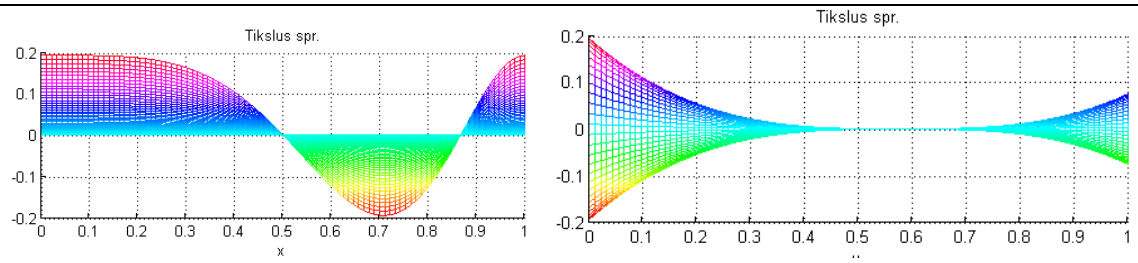
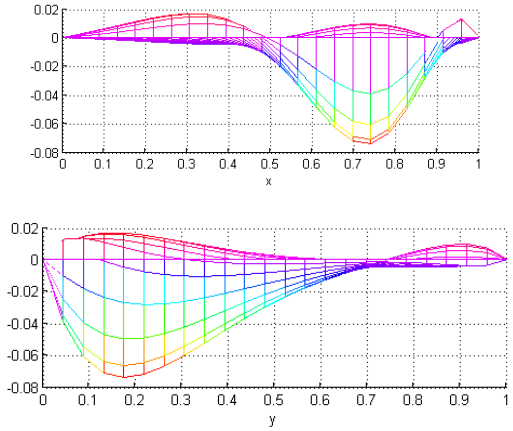
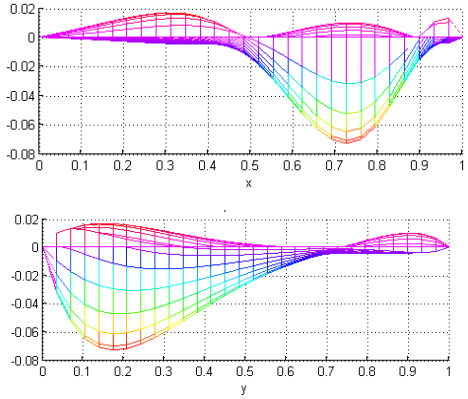
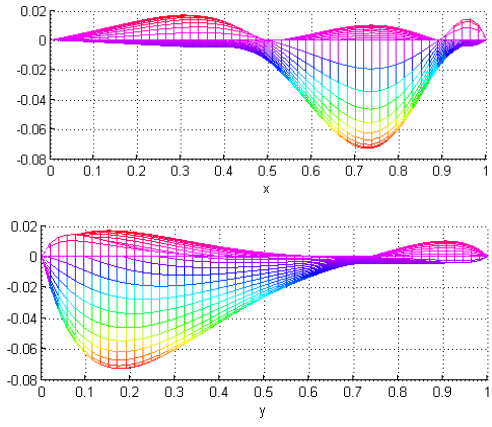
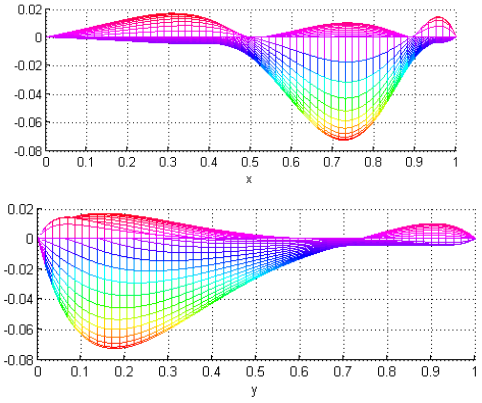
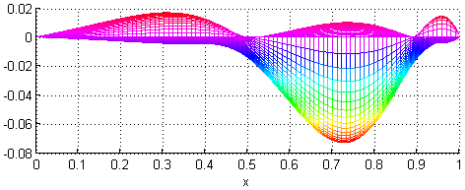
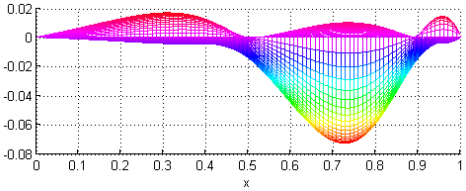
Gauname paklaidų priklausomybės grafikus kiekvienam uždaviniui (paklaida skaičiuojama kaip tikslaus ir apytikslio sprendinių skirtumo maksimumo norma) keičiant tiek schemos tikslumą, tiek žingsnį:

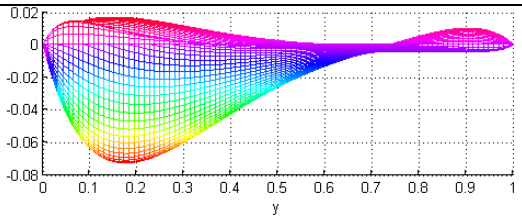
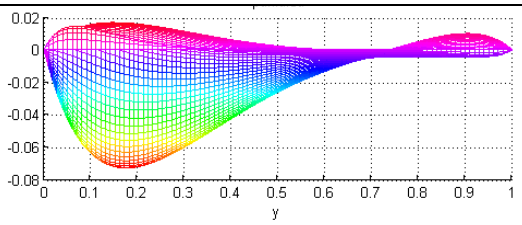
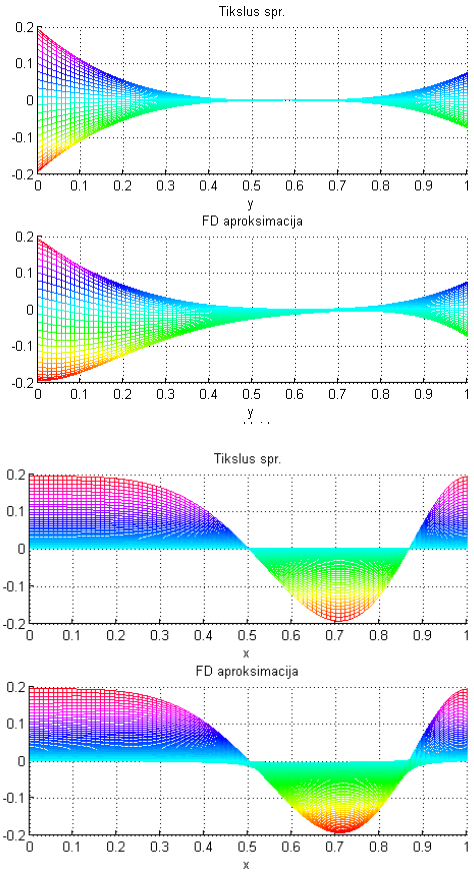
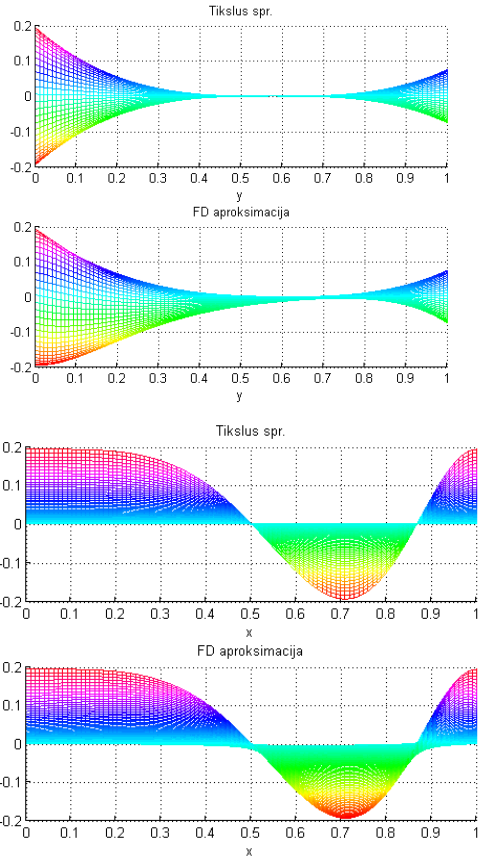
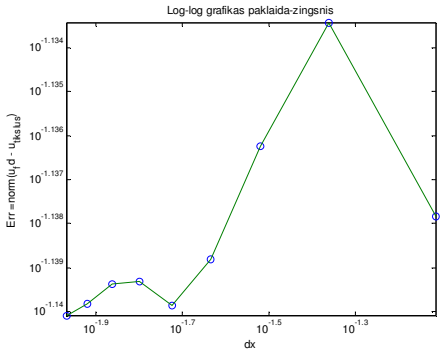
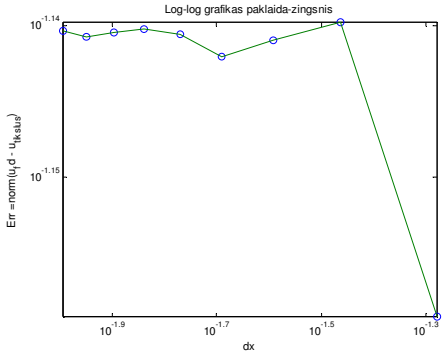


2.22 pav. 2D uždavinių paklaidų grafikai loglogaritminėje skalėje

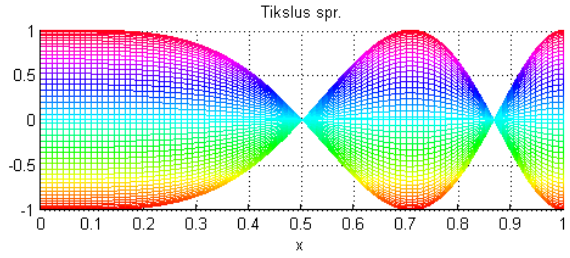
Matome, kad 1 ir 3 uždavinių paklados mažėja mažinant žingsnį ir didinant schemų tikslumą, ko ir tikėjomės pagal teoriją, tačiau 2 uždaviniui teorinės prielaidos nepasitvirtina. Panagrinėkime, kokio pavidalo paklaidos lieka 2 ir 3 uždaviniuose.

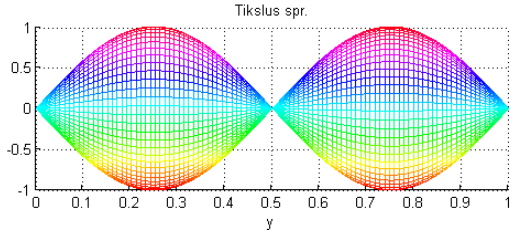
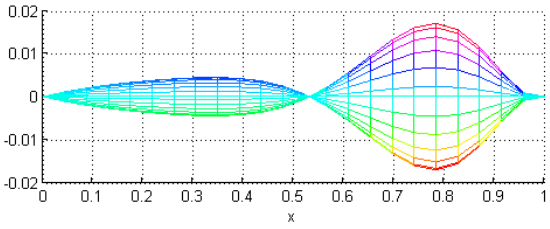
2.5 lentelė. 2D uždavinio nr.2 sprendinio aproksimavimo ir konvergavimo analizė esant kelioms tikslumo parametro m reikšmėms.

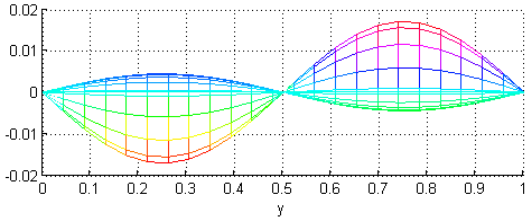
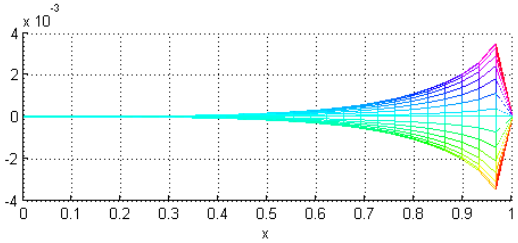
Sprendinio projekcijos x-z: y-z:		
n	2užd, paklaidos projekcija x-z, y-z	
	m=1	m=3
24, 26		
54, 56		
94, 96		

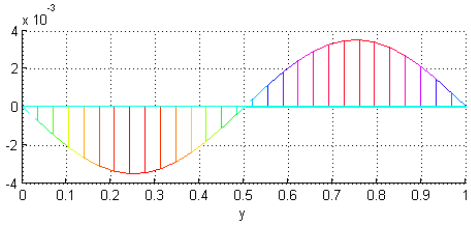
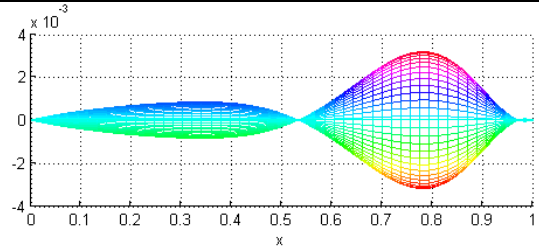
		
Gauto Sprendinio projekcijos su paskutiniu n		
paklaidos priklaus. nuo h		

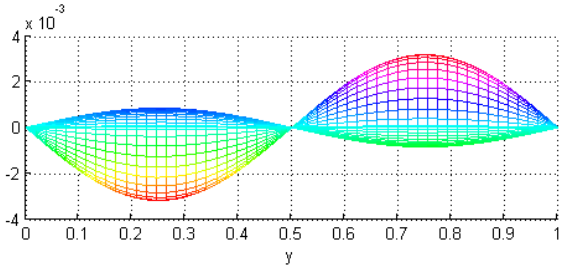
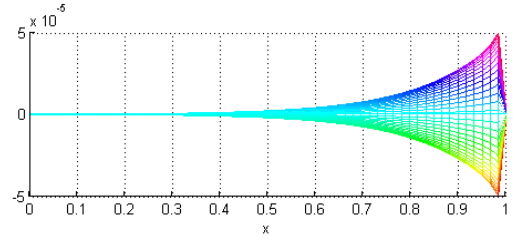
2.6 lentelė. 2D uždavinio nr.3 sprendinio aproksimavimo ir konvergavimo analizė esant kelioms tikslumo parametro m reikšmėms.

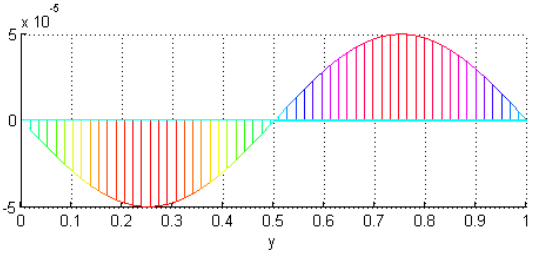
<table>
<tr><td>Sprendinio projekcijos
x-z:
y-z:</td><td colspan="2"> 

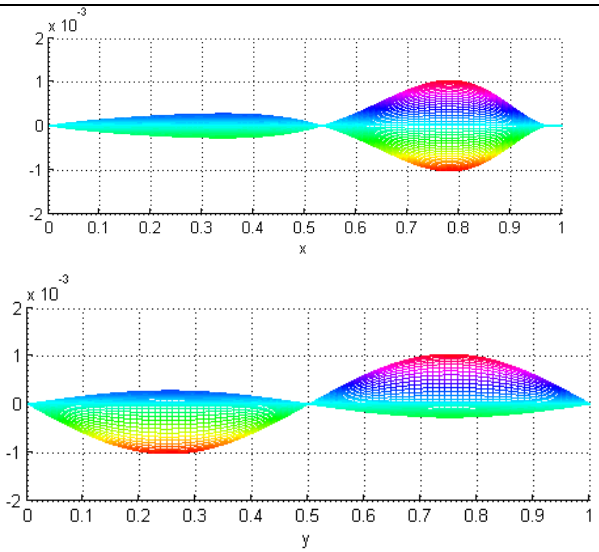
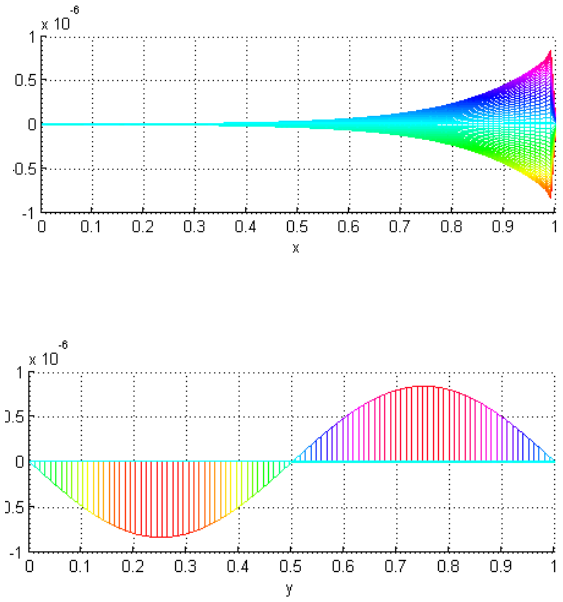
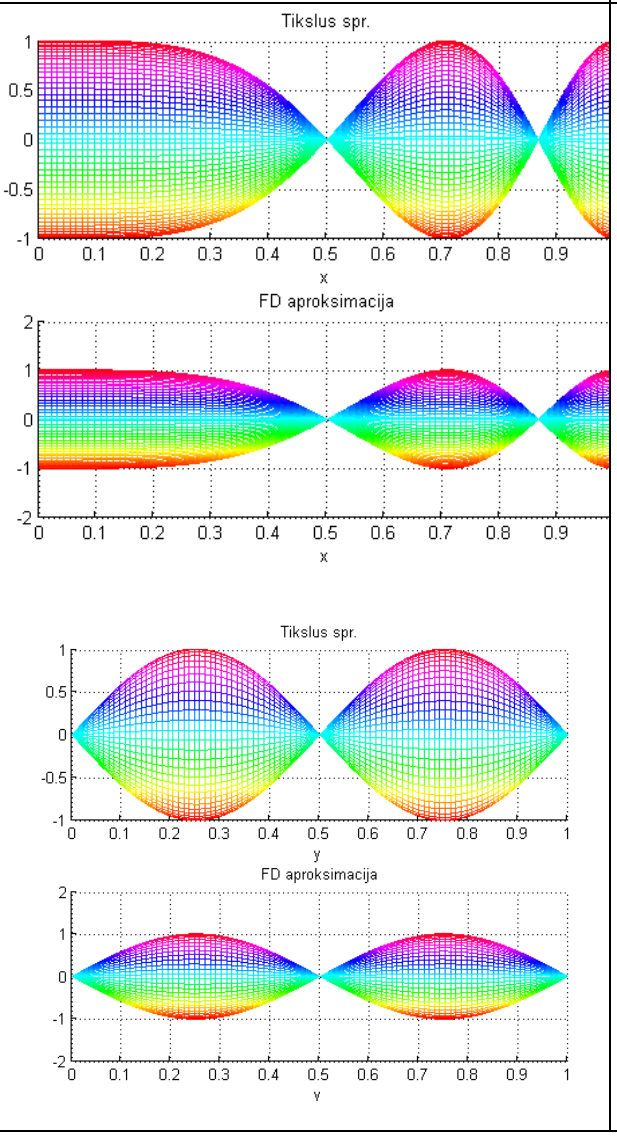
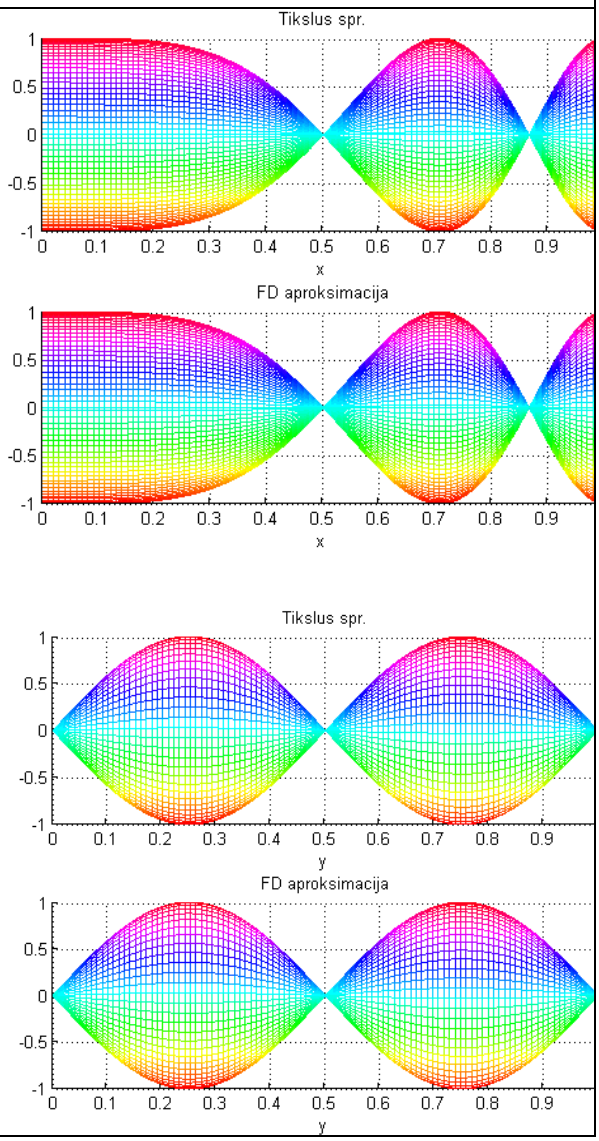

</td></tr>
<tr><td>n</td><td colspan="2">3užd, paklaidos projekcija x-z, y-z</td></tr>
<tr><td></td><td>m=1</td><td>m=3</td></tr>
<tr><td>24, 26</td><td> 

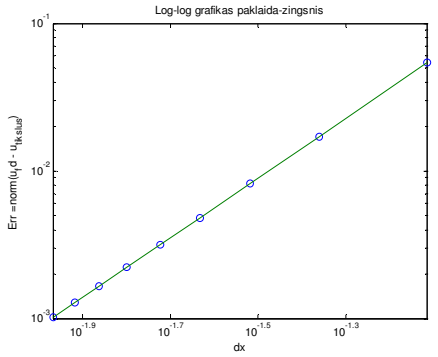
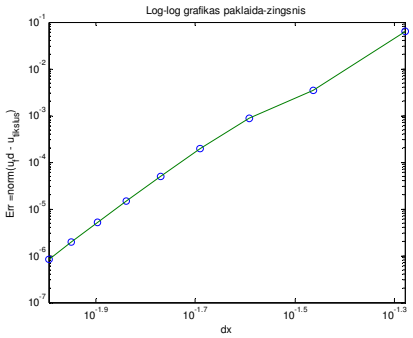

</td><td> 


</td></tr>
<tr><td>54, 56</td><td> 


</td><td> 

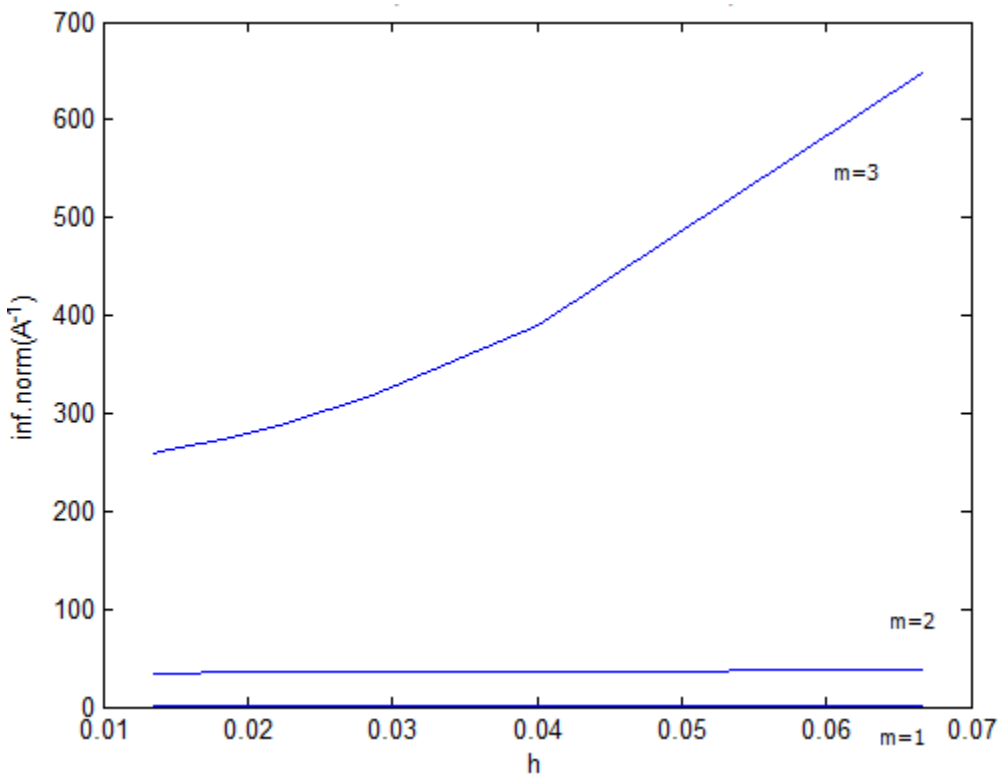

</td></tr>
</table>

94, 96		
Gauto sprendinio projekcijos su paskutiniu n		

Paklaidos priklaus. nuo h		

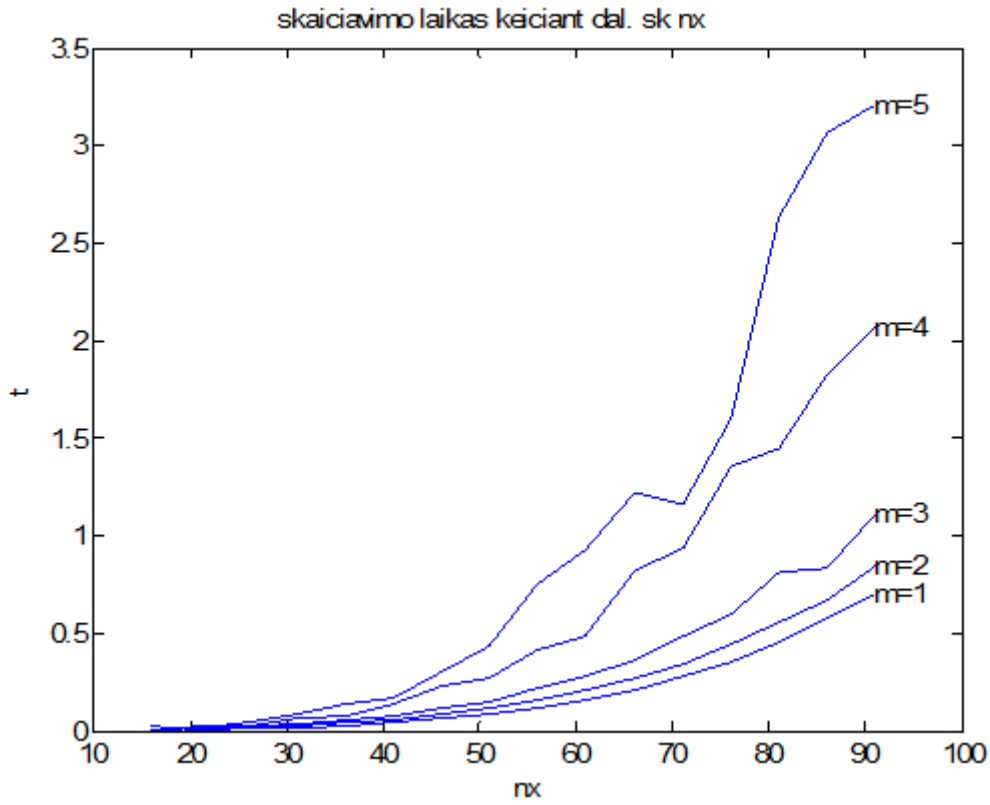
Iš 2-ojo uždavinio sprendinio aproksimacijų 2.5 lentelėje matyti, kad aproksimuotas sprendinys y-z projekcijoje yra išgaubtas, o tikslus – įgaubtas. Galima būtų atlikti platesnį tyrimą, ar tikslumą įtakoja sprendinio funkcijos įgaubtumas.

Nagrinėjamų schemų stabilumą patvirtina atvirkštinių matricų maksimumo normų sekos, kai žingsnis mažinamas – jos aprėžtos (šios sekos nepriklauso nuo to, kurį iš uždavinių nagrinėsime, nes koeficientai prie išvestinių pastovūs, todėl matricos sutampa visiems nagrinėtiems uždaviniams, kai žingsnis ir schemos tikslumo parametras fiksuoti):



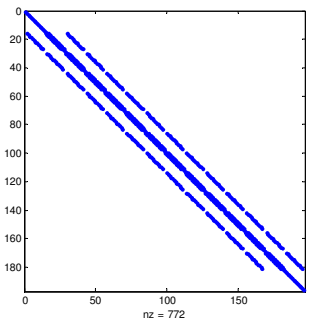
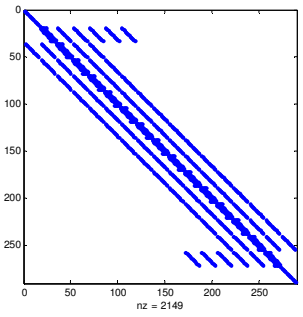
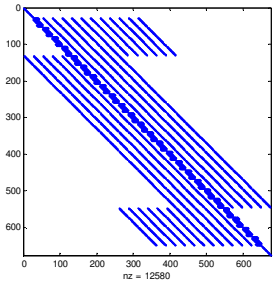
2.23 pav. 2D uždavinio schemų skaitinis stabilumo tyrimas

Toliau pateikiamas laiko grafikas įvairių tikslumų schemoms ir dalinimų skaičiams. Čia laikas nebe tiesiškai priklauso nuo dalinimų skaičiaus, kaip vienmačiu atveju, ir to buvo galima tikėtis, nes matricos dydis didėja kvadratiškai nuo dalinimų skaičiaus n



2.24 pav. 2D uždavinio sprendimo laiko priklausomybė nuo tikslumo parametro m ir tinklelio dalinimų išilgai vienos ašies skaičiaus nx

Ir kaip sudėtingėja matricos, kai m didinamas:

M=1	M=2	…	M=5
		…	

2.25 pav. 2D schemos matricos sudėtingumas esant įvairioms parametro m reikšmėms

Gauti rezultatai rodo, kad tikslinant schemą nebūtinai gausime atitinkamai geresnius rezultatus, tačiau galima tikėtis bent jau ne blogesnių rezultatų. 2 uždavinio sprendimo paklaidos beveik nesikeitė tiek tikslinant žingsnį, tiek schemą, o kitų nagrinėtų uždavinių paklaidos mažėjo mažinant žingsnį. Kadangi sistemos matricos dydis auga labai greitai smulkinant tinkelį, tai galime nagrinėti tik iki 100 dalinimo taškų, Dėl šios priežasties negalėjome pastebėti paklaidos didėjimo dar labiau smulkinant tinklelį, kaip stebėjome vienmačiu stacionariu atveju. Galima būtų daryti prielaidą, kad jei reikia tikslesnio psrendinio ir nėra labai brangus sprendimo laikas, dvimačiame stacionariame uždavinyje verta smulkinti tinklelį. Tikslinant schemas, 1 ir 3 uždaviniuose taip pat matėme, jog galima gauti mažesnes paklaidas, ir verta tikslinti maždaug iki m=4, nes žingsnio toliau mažinti negalime dėl kompiuterio operatyvinės atminties vienam kintamajam (sistemos matricai) saugoti ribojimui programoje, o grafikuose esant įvairių tikslumų schemoms paklaidų didėjimo mažinant žingsnį nepastebėta. 2 uždavinys išsiskyrė tuo, kad jis nejautrus nei žingsnio mažinimui, nei schemos tikslumo didinimui; čia problema tokia, kad neturėdami tikslaus sprendinio, naudodami didesnio tikslumo schemą tikėtumėmės geresnio tikslumo rezultato, nors realiai jo nepasiektume.

2.3.4. 1Dt atvejis

Sprendžiamas toks pradinio-kraštinio uždavinio atvejis su parametru α:

$$\begin{cases} \frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \ 0 < x < L, \ t > 0 \\ u(0,t) = 0, u(L,t) = 0, \qquad t > 0 \\ u(x,0) = f_0(x) \qquad 0 \le x \le 1 \end{cases} \tag{2.13}$$

Tokio uždavinio bendrasis sprendinys užrašomas begaline eilute:

$$u(x,t) = \sum_{n=1}^{\infty} b_n \sin\left(\frac{n\pi x}{L}\right) \exp\left(-\frac{\alpha n^2 \pi^2 t}{L^2}\right) \tag{2.14}$$

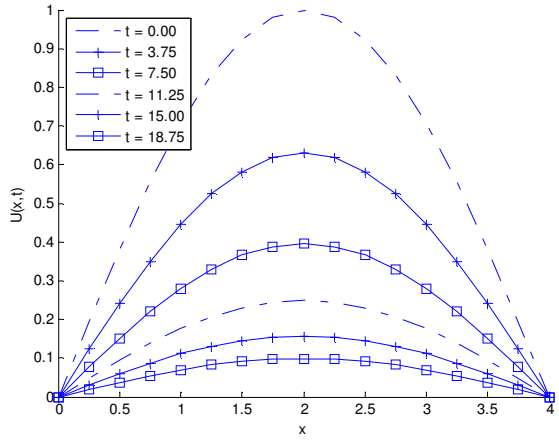
Čia $b_n = \frac{2}{L}\int_0^L f_0(x)\sin\left(\frac{n\pi x}{L}\right) dx$ (2.15)

Tuomet su pradine sąlyga: $u(x,0) = \sin\left(\frac{\pi x}{L}\right)$, pasinaudodami tuo, kad $\int_0^L \sin\left(\frac{\pi x}{L}\right)\sin\left(\frac{n\pi x}{L}\right) dx = L2$, kai n=10, kitu atveju, gauname sprendinį

$$u(x,t) = \sin\left(\frac{\pi x}{L}\right) \exp\left(-\frac{\alpha \pi^2 t}{L^2}\right). \tag{2.16}$$

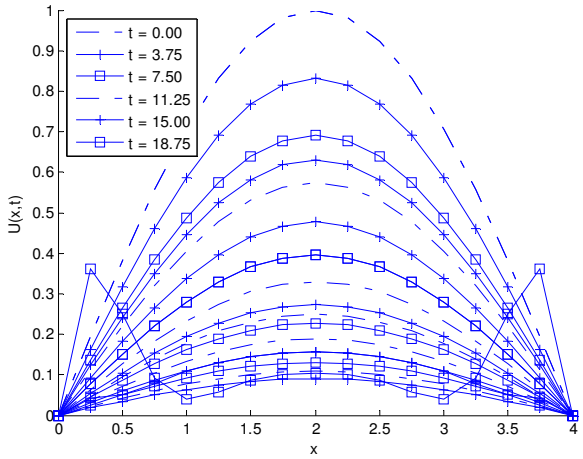
Matome, kad su teigiamu α sprendinys gęsta, t.y. strypui šilumos nesuteikiama laikui bėgant, o jo galuose temperatūra palaikoma lygi 0, todėl laikui bėgant šiluma sklaidosi į aplinką. Pagal šią savybę matysime, kada metodas nestabilus ir įvertinsime su kuriais tikslumo parametrais p ir m nagrinėjama schema yra stabili.

Sprendinio grafikas įvairiais laiko momentais:



2.26 pav. 1Dt uždavinio sprendiniai įvairiais laiko momentais

Šį uždavinį sprendėme dviem būdais. Pirmas - daugiažingsnis skirtuminių schemų metodas; jam inicializuoti panaudojome ekstrapoliaciją. Tolesnius žingsnius skaičiuojame jau pagal pateiktą schemą metodų skyriuje. Antras - metodas su vienažingsne schema laiko atžvilgiu, tikslumą t atžvilgiu išgaunant tik naudojant ekstrapoliaciją. Kadangi pagal teoriją naudojamos $O(k^2)$ tikslumo kairinio laiko - centrinės erdvės schema yra besąlygiškai stabili, tai teoriškai, kad ir kokius t ir h parinktume, ekstrapoliuotas sprendinys taip pat turi būti stabilus (ekstrapoliacijai naudojama vieno laiko žingsnio skirtuminė schema). Pirmojo metodo su aukštesnėmis tikslumo eilėmis stabilumui trūksta teorinių nagrinėjimų. Kad pastebėtume nestabilumą, simuliavimo laikas turi būti pakankamai ilgas, nes pirmuose žingsniuose ir nestabilus metodas nesukaupia pastebimai daug paklaidų. Nagrinėkime laiką tmax=20. Pavydžiui, su parametrais dx=0,25, t=0.25, p=6, m=2:

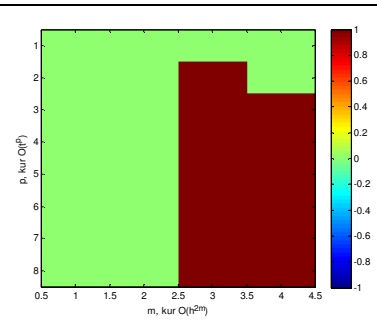
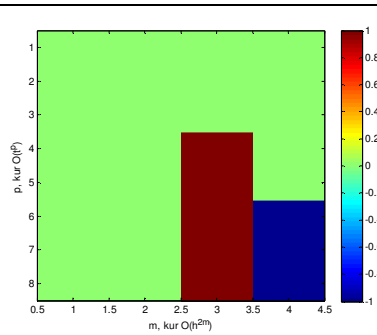
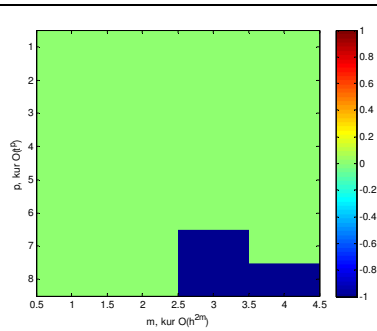
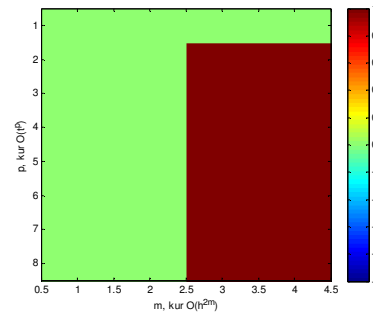
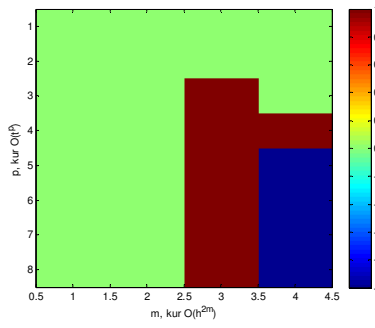
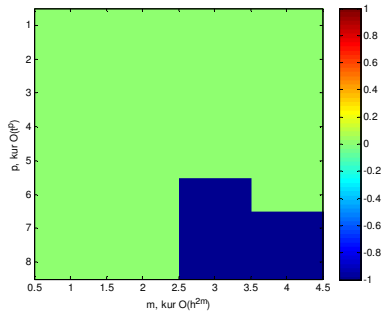
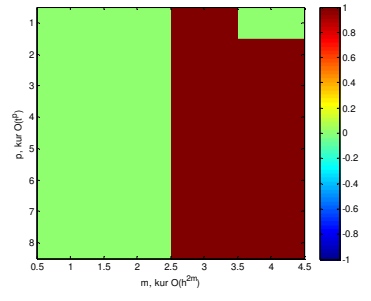
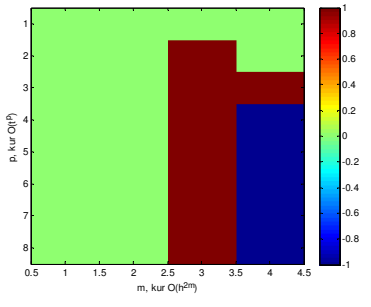
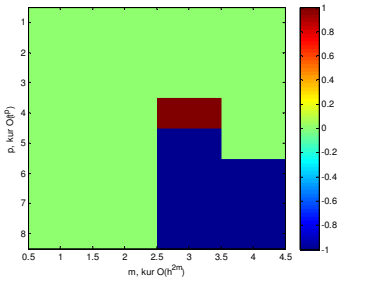


2.27 pav. 1Dt uždavinio schemos nestabilumas

Parinkę įvairius žingsnių parametrus dx ir dt, gauname konvergavimo sritis (-1 reiškia, kad paklaidos nebuvo galima apskaičiuoti, nes skaičiai viršija kompiuterio limitus, o +1 reiškia, kad paklaida buvo didesnė už 1 – abu šie atvejai reiškia metodo nekonvergavimą).

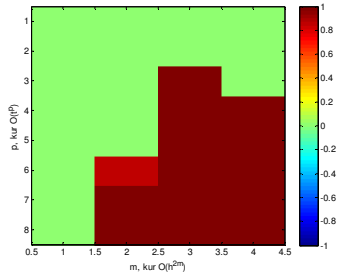
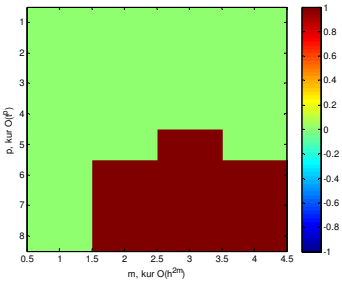
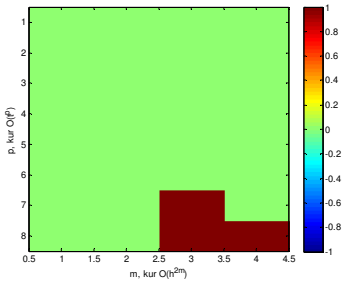
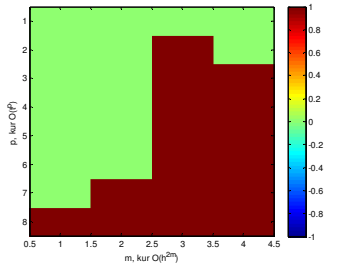
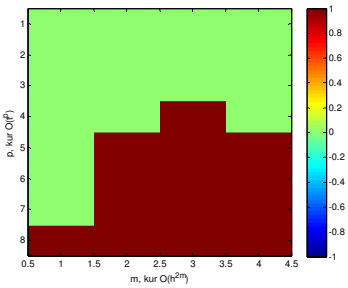
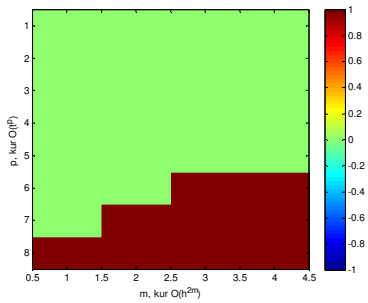
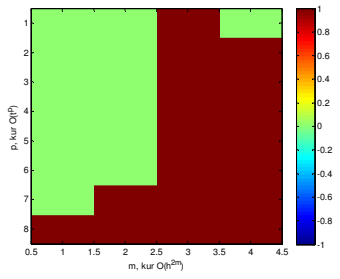
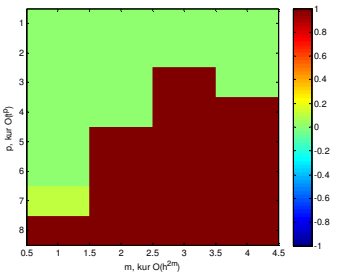
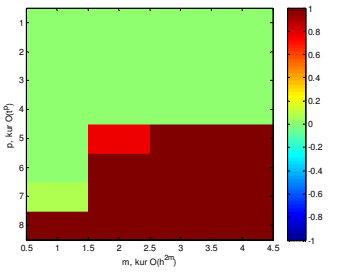
Iš konvergavimo grafikų 2.7 lentelėje matome, kad esant mažoms parametro m reikšmėms su visais nagrinėjamais p schemos konverguoja, ko ir tikėjomės iš ekstrapoliacijos metodo, kuriame naudota vienažingsnė laiko schema, kuri yra besąlygiškai stabili , kai m=2, p=1. Matome, kad iš tikrųjų, kaip ir buvo galima tikėtis, su m=2 schema stabili visiems p, ekstrapoliavus iš p reikšmių, apskaičiuotų su p=1. Tačiau su didelėm parametro m reikšmėm schemos gali ir nekonverguoti

2.7 lentelė. 1Dt uždavinio sprendinio konvergavimo sritys esant kelioms laiko ir erdvės žingsnių reikšmėms, tikslumui pasiekti naudojant ekstrapoliaciją.

Metodas tik ekstrapoliuojant iki norimo tikslumo p			
dx-> dt	0.25	0.125	0.0625
0.25			
0.125			
0.0625			

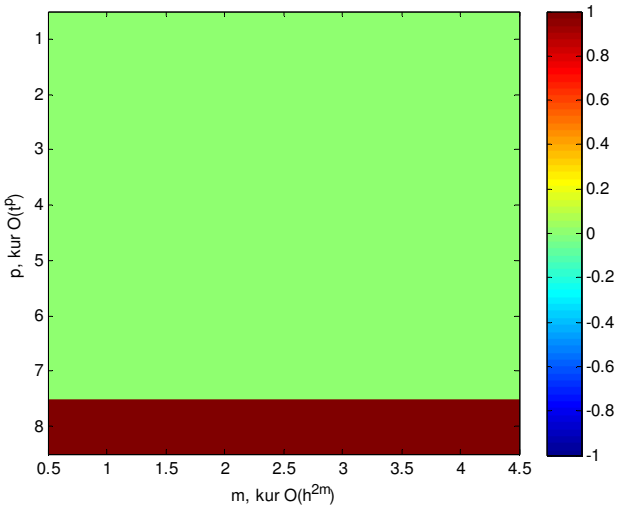
Kaip vienmačiame stacionariame uždavinyje pastebėjome, didesnio tikslumo schemos su dideliais žingsniais (retais tinkleliais) gali duoti dideles paklaidas (nors ir yra stabilios). Ir matome iš konvergavimo sričių grafikų, kad mažinant dx, nekonvergavimo sritis mažėja (žiūrėti lentelėje pagal eilutes: išilgai abscisių ašies- parametro m reikšmės, išilgai ordinačių ašies – parametro p reikšmės).

2.8 lentelė. 1Dt uždavinio sprendinio konvergavimo sritys esant kelioms laiko ir erdvės žingsnių reikšmėms, ekstrapoliaciją naudojant tik daugiažingsnio metodo inicializacijai.

Metodas su ekstrapoliacija inicializavimui, daugiažingsnis			
dx-> dt	0.25	0.125	0.0625
0.25			
0.125			
0.0625			

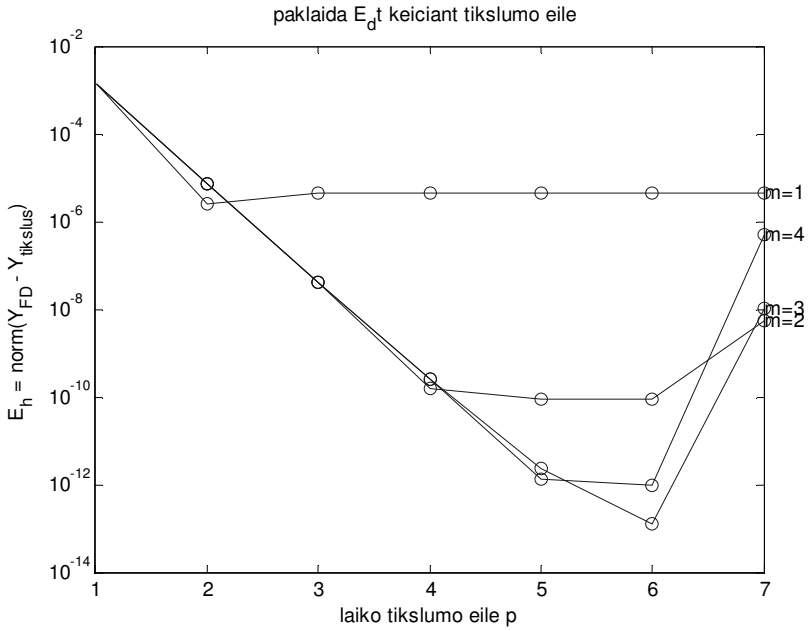
Kaip ir metode vien su ekstrapoliacija, didelių tikslumų schemoms reikia smulkesnio tinklelio – mažinant dx, esant fiksuotam laiko žingsniui, konvergavimo sritis didėja. Patikriname, ar gautų schemų tikslumai atitinka teorinį konvergavimo greitį. Nagrinėjame didesnių tikslumų skirtuminių schemų

metodą. Fiksuojame mažą reikšmę dx, pvz dx=0.0625/4, ir tmax=10, dt=0.0625, tuomet konvergavimo sritis:



2.28 pav. 1Dt daugiažingsnio metodo su dx=0.0625/4, ir tmax=10, dt=0.0625 konvergavimo sritis

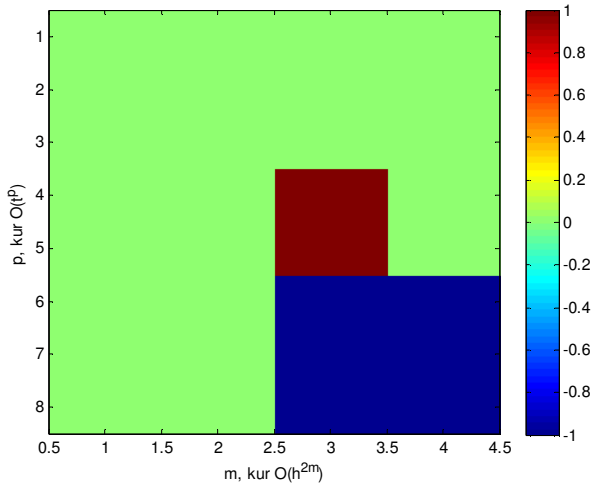
Matome, kad su šiais parametrais, konvergavimo sritis [m,p]=[1:4, 1:7]. Fiksavus tikslumo parametrą m erdvės kintamojo atžvilgiu m=1,…,4, gauname, kaip paklaida kinta keičiant laiko kintamojo tikslumo parametrą p. Kadangi schemos tikslumas teoriškai yra $O(dx^{2m}) + O(dt^{p})$, tai paklaida turėtų keisti tikslumo eilę iki kol p=2m, po to teoriškai kitimas turėtų būti nepastebimas.



2.29 pav. 1Dt daugiažingsnio metodo paklaidos pastebimai mažėja iki kol laiko ir erdvės kintamųjų tikslumo eilės susilygina: p=2m

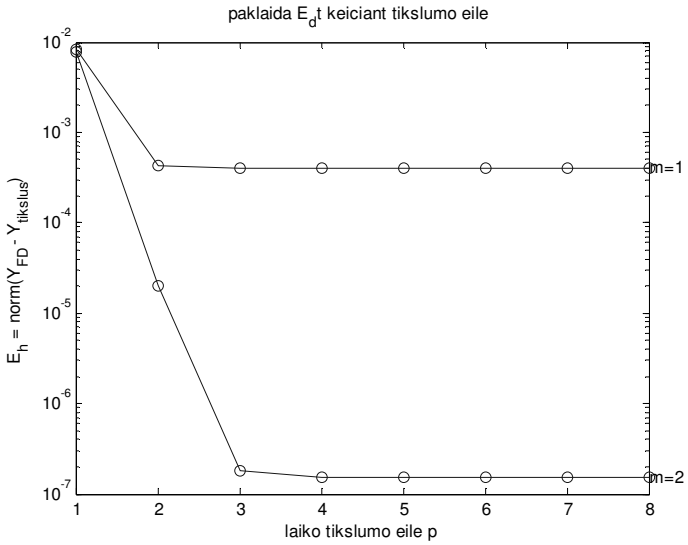
Gautasis grafikas rodo, kad iš tikrųjų, paklaida mažėjo iki kol p=2m, tačiau toliau didinant p, paklaida didesnių tikslumų schemose išaugo. Iš šio pavyzdžio galima būtų daryti prielaidą, kad kaip ir teoriškai, taip ir šiame metode verta imti p=2m.

Analogiškas tyrimas su pilnai ekstrapoliaciniu metodu; fiksuojame reikšmę dx, pvz dx=0.0625, ir tmax=10, dt=0.0625, tuomet konvergavimo sritis:



2.30 pav. 1Dt schemos ekstrapoliacinio metodo su dx=0.0625/4, ir tmax=10, dt=0.0625 konvergavimo sritis

Gaunama konvergavimo sritis [m,p]=[1:2, 1:8]. 2.31pav. grafikas atkartoja teorinius samprotavimus: iki p=2m paklaidos eilė mažėja, vėliau beveik nekinta dėl didesnės paklaidos dedamosios $O(h^{2m})$.



2.31 pav. 1Dt ekstrapoliacinio metodo paklaidos pastebimai mažėja iki kol laiko ir erdvės kintamųjų tikslumo eilės susilygina: p=2m

Fiksavę pakankamai mažą dx bei keisdami dt, skaičiuojame konvergavimo greičio parametrą p, pasinaudodami apytikslia lygybe

$$\frac{TP_2}{TP_1} = \frac{K_t\left(\frac{dt}{2}\right)^p + K_x dx^{2m}}{K_t dt^p + K_x dx^{2m}} = (dx \ll dt) \approx \left(\frac{1}{2}\right)^p. \qquad (2.17)$$

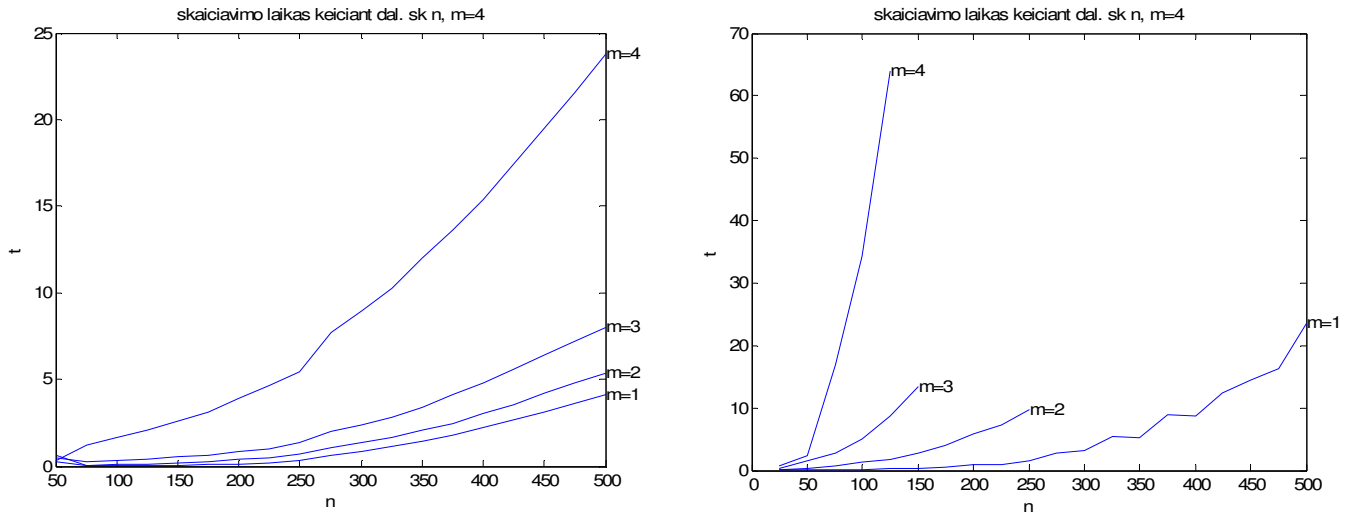
Imdami dx=0.0625/8, dt=0.0625*16, tmax=20 ir imdami 5 skaičiavimo iteracijas dt:=dt/2 (dx-fiksuotas, m fikuotas: m=ceil(p/2)) gauname tokius konvergavimo greičio įverčius :

2.9 lentelė. 1Dt uždavinio sprendinio konvergavimo greičių palyginimas esant ekstrapoliaciniam metodui ir daugiažingsniam metodui.

p	ekstrapoliacinio	schemų
1	0.997	0.997
2	1.842	2.057
3	2.917	3.009
4	4.304	4.010
Didesniems p su parinktais parametrais toliau mažinant dt, paklaidos ima didėti.		

Pirma eilutė sutampa, nes kai p=1, metodai sutampa – naudojamas vienažingsnis BTCS metodas.

Pagal metodų skaičiavimo eigą aišku, kaip ir vienmačiame stacionariame uždavinyje, kad laiko sąnaudos pilnai ekstrapoliaciniam sprendimui yra didesnės, nes tas pats uždavinys sprendžiamas pakartotinai tuo daugiau kartų, kuo didesnio tikslumo siekiama. Laiko grafikams gauti imame h:=dx=dt ir p=2m, tmax=4.



2.32 pav. 1Dt uždavinio sprendimo laikai sprendžiant a) daugiažingsniu ir b) ekstrapoliaciniu metodu.

IŠVADOS

Iš nagrinėtų pavyzdžių aišku, kad nėra bendrų taisyklių optimaliam skirtuminės schemos pasirinkimui: ar pasitvirtins teoriškai laukiamas rezultatas, priklauso nuo sprendinio specifikos. Todėl gavus uždavinį, prieš renkantis skirtuminę schemą bei žingsnį, reikia atlikti jo tyrimą. Tinkamai parinkus schemos parametrus, aukštesnių tikslumų skirtuminėmis schemomis galima gauti tikslesnius rezultatus.

Mažesnio tinklelio žingsnio pasirinkimas ir didesnio tikslumo schemos pasirinkimas nebūtinai pakeičia vienas kitą. Yra uždavinių, kuriems parinkus netinkamą žingsnį ir didesnio tikslumo schemą, gaunama dar didesnė paklaida. Aptartas būdas, kaip tą pastebėti sprendžiant realų uždavinį.

Parabolinio tipo pradiniam-kraštiniam uždaviniui įgyvendinta ekstrapoliacijos idėja didesnio tikslumo skirtuminėms schemoms inicializuoti; tai leido pasiekti norimą schemos tikslumo eilę. Konvergavimo sritims patikrinti pasiūlytas vizualinis būdas, pagrįstas nagrinėjamo uždavinio sprendinio savybėmis.

Rekomendacijos tolesniam tyrimui:

Galima ieškoti sąsajų tarp sprendinio savybių ir schemų parametrų tinkamumo.

Šiame darbe paprasti schemų sudarymo metodai adaptuoti didesnių tikslumų schemoms sudaryti, nesiekiant didžiausio efektyvumo. Galima ieškoti efektyvių algortitmų schemoms su pasirinktais - konkretizuotais- parametrais realizuoti.

LITERATŪRA

1. R.J. Le Veque. Finite Difference Methods for differenctial equations. University of Washington, 2006. 144 p.
2. G. W. Recktenwald. Finite-Difference Approximations to the Heat Equation. 2011. -27p.
3. J. C. Strikwerda. Finite difference schemes and partial differential equations. Siam. -448p.
4. D Eberly. Derivative Approximation by finite differences. www.geometrictools.com., 2008. -7p.
5. R. Čiegis. Diferencialinių lygčių skaitiniai sprndimo metodai. Vilnius, 2003. -434p.
6. K. Plukas. Skaitiniai metodai ir algoritmai. Kaunas, 2001. -549p.
7. G. S. Dosinas, L. Papreckienė. Diferencialinės lygtys.KTU, Kaunas, 2005. -222p.
8. Pekarskas V. Diferecialinis ir integralinis skaičiavimas. 2 dalis. KTU. -Kaunas, 2003. -417p.
9. en.wikipedia.org: [finite difference coefficient, Neumann stability, heat equation, boundary value problem, boundary conditions, etc.]
10. A. A. Samarskii. The theory of difference schemes. Moscow, 2001. -788p.
11. V. G Ganzha. Computer-aided analysis of difference schemes for partial differential equations. New York, 1996. -473p.
12. I. P. Gavrilyuk, M. Hermann, V. L. Makarov, M. V. Kutniv. Exact and truncated difference schemes for boundary value odes. Birhauser, 2011. -260p.

PRIEDAI

1 priedas. Kai kurios 2.3.1 skyriaus programos

Skirtuminių santykių koeficientų apskaičiavimas *Matlab*:

```
function C=koeficientai(pts,order,bcf)
%pts- points - per kiek tasku skirtuminis santykis
%order - isvestines eile
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% h     - atstumas tarp mazgu parenkamas 1, tuomet gaunami koeficientai
% skirtumams kuriuos dar reikia padalinti is atitinkamo h laispnio
% pts   - tasku, per kuriuos skirtuminio santykio ieskom, skaicius
% pts=tikslumo eile + order
% order - isvestines eile order<pts-1
% bcf = backward jei -1, central jei 0(arba else nei -1 ar 1) ir forward jei 1
switch bcf
    case 1 % forward
        imin=0;
        imax=pts-1;
    case -1
        imin=-(pts-1);
        imax=0;
    otherwise
            imin=-floor((pts-1)/2);
            imax=-imin;
end
ivekt=(imin:1:imax);
A=ones(pts,pts);
for i=1:pts-1
      A(i+1,:)= ivekt.^i;
end
B=zeros(pts,1);
B(order+1)=1;
C=A\B;
C=C.*factorial(order);
C=C.';
```

Vienas iš tyrimo metodų:

```
% pasirenkama tiriamos isvestines eile N, centrinio, kairinio/desininio
% skirt sant aproksimaciju eiles, ir keiciant zingsni pavaizduojamos
% isvestiniu pakeitimo skirt santykiais paklaidos
clc
clear all
kiek=28; % kiek skirtingu zingsniu tarp 10^-1 ir 10^-16
h = logspace(-0.5,-10,kiek);
x0 = 10;
% N - isvestines eile
N=1;
m=1;  % centrinio skirtuminio santykio tikslumo eile m=1 O(h^2), m=2 O(h^4) ir t.t. O(h^2m)
m2=2; % kairinio ir desininio sk sant tikslumas O(h^m)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Nr=2;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
switch Nr
    case 1
        fja = @(x) 2.*x+4./x;
        syms x;
        isvestine= matlabFunction(diff(2.*x+4./x,N));
    case 2
        fja= @(x) cos(x);
        syms x;
        isvestine= matlabFunction(diff(cos(x),N));
    case 3
```

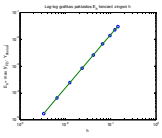
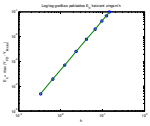
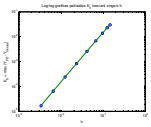
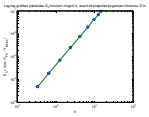
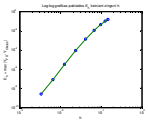
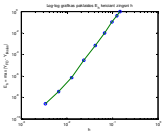
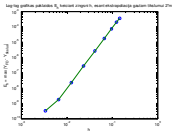
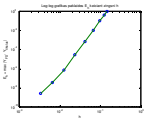
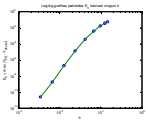
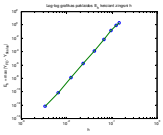
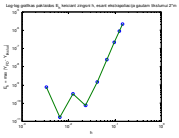
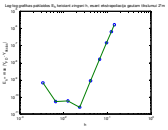
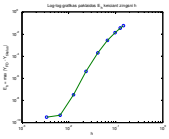
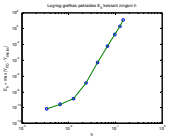
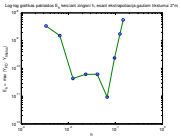
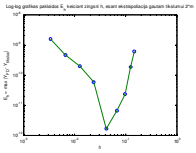
```
        syms x;
        isvestine= matlabFunction(diff(cos(6*x),N));
        fja=@(x) cos(6*x);
end
% kai tikslumas O(h^2m), tai central approx reikia 2m+1 tasku, taigi koef
% vektorius bus 2m+1 ilgio
lyg = (mod(N,2)==0); % =1 jei  N lyginis, =0 jei N nelyginis
koef1 = koeficientai(2*m+N-lyg, N, 0); % f^(N) skirt santykio central koef
% forward ir backward koef.:
koeff1 = koeficientai(m2+N, N, 1); % f^(N) skirt santykio forward koef
koefb1 = koeficientai(m2+N, N, -1); % f^(N) skirt santykio backward koef
yDc1=zeros(1,kiek);
c=ceil((2*m+N-lyg)/2);
mm=floor((2*m+N-lyg)/2);
 for ii=-mm:mm
     yDc1=yDc1+ fja(x0+ii*h).*koef1(1,c+ii);
 end
yDc1=yDc1./(h.^N);
yDf1=zeros(1,kiek);
yDb1=zeros(1,kiek);
for ii=0:(m2+N-1)
     yDf1=yDf1+ fja(x0+ii*h).*koeff1(1,ii+1);
     yDb1=yDb1+ fja(x0-ii*h).*koefb1(1,m2+N-ii);
end
yDf1=yDf1./(h.^N);
yDb1=yDb1./(h.^N);
yDe = isvestine(x0);
eDf = abs(yDf1-yDe);
eDb = abs(yDb1-yDe);
eDc = abs(yDc1-yDe);

figure(1)
loglog(h,eDc,'r',h,eDb,'b--',h,eDf,'g:');
nr=find(diff(eDc)>0, 1,'first'); % kada ji keicia zenkla, ima dideti, t.y. tampa teigiama-
pazymet kvadraciuka, iki to tinka pagal teorija
hm=h(nr);
hold on;
st = sprintf(['O(h^',num2str(2*m),') centr.']);
legstr = st;
loglog(hm,eDc(nr),'s')
nr=find(diff(eDb)>0, 1,'first');
hm=h(nr);
hold on;
st = sprintf(['O(h^',num2str(m2),') kair.']);
legstr=strvcat(legstr,st);
loglog(hm,eDb(nr),'s')
nr=find(diff(eDf)>0, 1,'first');
hm=h(nr);
hold on;
st = sprintf(['O(h^{',num2str(m2),'}) dešin.']);
legstr = strvcat(legstr,st);
loglog(hm,eDf(nr),'s')
legend(legstr,2)
xlabel('h');
ylabel('|aproksimavimo skirt. santykiu paklaida|');
title('Log-log: isvestines aproksimavimo skirtuminiais santykiais paklaida keiciant žingsni
h')
set(gcf,'Color',[1 1 1]);
```

2 priedas. 2.3.2 skyriaus papildoma medžiaga

2.1. Konvergavimo greičio įverčių palyginimas

Ekstrapoliacijos būdu ir aukštesnių eilių schemomis gautų sprendinių konvergavimo greičių įverčiai per 9 iteracijas žingsniu $2 * 2^k + 21$, k=1,..,9.

Siekiamo tikslumo eilė	Aukštesnių eilių schemos		ekstrapoliacija	
p=2m	Už 1	Už3	Už1	Už3
2	1.973 	2.024 	1.973 	2.024 
4	4.396 	5.365 	3.730 	4.035 
6	5.124 	7.474 	3.125 	3.171 
8	4.782 	7.915 	1.788 	1.368 

2.2. Kai kurios 2.3.2 skyriaus programos

Paklaidų skaičiavimas, keičiant žingsnį ir tikslumo eilę:

```
clc
clear all
%  a1(x) d/dx(dy/dx) + a2(x) dy/dx + a3(x) y + a4(x) = 0
Nr=3;
switch Nr
case 1
%-------------------------------------------------------------
% 1.
a1=inline('1','x') ;
a2=inline('1/x','x') ;
a3=inline('-1/x^2','x') ;
a4=inline('0','x') ;
```

```
% krastines reiksmes
x0=0.5 ;
y0=9 ;
xf=4 ;
yf=9 ;
sprendinys = @(x) 2.*x+4./x;
%-------------------------------------------------------------
case 2
%-------------------------------------------------------------
% 2.
a1=inline('-exp(x)','x') ;
a2=inline('-exp(x)','x') ;
a3=inline('x^3','x') ;
a4=inline('-(x^3)*cos(x)-exp(x)*(sin(x)+cos(x))','x') ;

sprendinys = @(x) cos(x);
% krastines reiksmes
x0=0.5 ;
y0=sprendinys(x0) ;
xf=4 ;
yf=sprendinys(xf) ;
%-------------------------------------------------------------
case 3
%-------------------------------------------------------------
% 3.
a1=inline('-exp(x)','x') ;
a2=inline('-exp(x)','x') ;
a3=inline('x^3','x') ;
a4=inline('-(x^3)*cos(6*x)-exp(x)*(6*sin(6*x)+(6^2)*cos(6*x))','x') ;

sprendinys = @(x) cos(6*x);
% krastines reiksmes
x0=0.5 ;
y0=sprendinys(x0) ;
xf=4 ;
yf=sprendinys(xf) ;
%-------------------------------------------------------------
case 4
%-------------------------------------------------------------
% 3.
a1=inline('-exp(x)','x') ;
a2=inline('-exp(x)','x') ;
a3=inline('x^3','x') ;
a4=inline('-(x^3)*cos(24*x)-exp(x)*(24*sin(24*x)+(24^2)*cos(24*x))','x') ;

sprendinys = @(x) cos(24*x);
% krastines reiksmes
x0=0.5 ;
y0=sprendinys(x0) ;
xf=4 ;
yf=sprendinys(xf) ;
%-------------------------------------------------------------
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mmax=5;
hhh=zeros(mmax,1);
ttt=zeros(mmax,1);
sumt=zeros(mmax,1);
for m=1:mmax
%m=7;  % tikslumo eile m=1 O(h^2), m=2 O(h^4) ir t.t. -> max m = 8, toliau skirtumu koef
apskaiciuojami netiksliai

koef1 = koeficientai(2*m+1, 1, 0); % f^(1) skirt santykio central koef
koef2 = koeficientai(2*m+1, 2, 0); % f^(2) skirt santykio central koef
if m>1 % tuomet reikia forward ir backward koef.
```

```
  koeff1 = koeficientai(2*m+1, 1, 1);  % f^(1) skirt santykio forward koef
  koefb1 = koeficientai(2*m+1, 1, -1); % f^(1) skirt santykio backward koef
  koeff2 = koeficientai(2*m+2, 2, 1); % f^(2) skirt santykio forward koef
  koefb2 = koeficientai(2*m+2, 2, -1); % f^(2) skirt santykio backward koef
end
%*********************************************************************
% tiriama paklaidos priklausomybe nuo dalinimu skaiciaus
kmax=9;
sz = kmax;
T=zeros(sz,1);
RMSE=zeros(sz,1);
laikas=zeros(sz,1);
MaxPakl=zeros(sz,1);
H=zeros(sz,1);

%*********************************************************************
for k=1:kmax-m

    n = 2*2^k + 21;        %  pradinis (reikiamas apskaiciuoti norimu tikslumu)tinklelio tasku
skaicius
    h=(xf-x0)/(n-1)        % pradinis zingsnis, su kuriuo skaiciuojama m=1 tikslumo aprox, su
h/2 skaiciuojama 2k aukstesnes eiles aprox

     clear A B Y x;
% zingsnis
h=(xf-x0)/(n-1) ;
% lygciu sistemos sudarymas
for i = 1:n
 x(i)=x0+(i-1)*h;
end
A=zeros(n,n);
B=zeros(n,1);
B(1)=y0;
B(n)=yf;
A(1,1)=1;
A(n,n)=1;
c=m+1; % centrinio elemento pozicija koef vektoriuje
% % O(h), centrinis antrai ir desininis pirmai
iki= n-m;
dum=2*m;
ilgis=2*m+1;
ilgis2=ilgis+1;
tstart=tic;
% kai krastuose nepakanka tasku centriniam skirt, naudojam forward prdzioj,
% per 2m+1 taskus f^(1) ir 2m+2 taskus f^(2)
for i=2:m
    A(i,i)  = a1(x(i))*koeff2(1,1)/h/h+a2(x(i))*koeff1(1,1)/h+a3(x(i)); % diagonalinis
elementas
    B(i)=-a4(x(i));
    for ii=1:dum
      A(i,i+ii)= a1(x(i))*koeff2(1,1+ii)/h/h+a2(x(i))*koeff1(1,1+ii)/h;
    end
    A(i,i+ilgis)= a1(x(i))*koeff2(1,ilgis2)/h/h; %tik f^(2) skirtum santykio koef
end

% kur centriniai skirt santykiai jau gali buti taikomi aproksimacijai, t.y.
% pakanka elementu i prieki ir atgal:
for i=c:iki
    A(i,i)  = a1(x(i))*koef2(1,c)/h/h+a2(x(i))*koef1(1,c)/h+a3(x(i)); % centrinis elementas
m+1
    B(i)=-a4(x(i));
    for ii=1:m
      A(i,i-ii)= a1(x(i))*koef2(1,c-ii)/h/h+a2(x(i))*koef1(1,c-ii)/h;
      A(i,i+ii)= a1(x(i))*koef2(1,c+ii)/h/h+a2(x(i))*koef1(1,c+ii)/h;
    end
```

```
end
% ir backward gale:
for i=iki+1:n-1
    A(i,i)  = a1(x(i))*koefb2(1,ilgis2)/h/h+a2(x(i))*koefb1(1,ilgis)/h+a3(x(i)); %
diagonalinis elementas
    B(i)=-a4(x(i));
    for ii=1:dum
      A(i,i-ii)= a1(x(i))*koefb2(1,ilgis2-ii)/h/h+a2(x(i))*koefb1(1,ilgis-ii)/h;
    end
    A(i,i-ilgis)= a1(x(i))*koefb2(1,1)/h/h; %tik f^(2) skirtum santykio koef
end

A=sparse(A);
Y=A\B;

laikas(k,1) = toc(tstart);
Pakl(k,1)=norm(Y-sprendinys(x).',inf);
sumt(m)=sumt(m)+laikas(k,1);
H(k,1)=h;
end
disp(m)
disp(k)
laikas
[RMSEmin, nr]=min(Pakl);
hmin=H(nr,1);
hhh(m)=RMSEmin;
ttt(m)=laikas(nr,1);

kk=1/(2*m+2);
loglog(H,Pakl, 'Color', [kk mod(m,2)==1 mod(m,2)==0]);
hold on;
loglog(hmin,RMSEmin,'s','Color', 'r')
hold on;
text(hmin/1.5,RMSEmin/2,['O(h^{',num2str(2*m),'})'] )
hold on;
xlabel('h');
ylabel('paklaidos inf. norma');
title(['paklaida keiciant h, kai m =1:1:', num2str(mmax)]);
set(gcf,'Color',[1 1 1]);
 end
```

Ekstrapoliacijos koeficientų apskaičiavimas *Matlab*:

```
function C=koeficientaiExtra2(k)
% randami koeficientai extrapoliavimui, k - koeficientu vektoriaus ilgis,
% tuo paciu ir puse norimos tikslumo eiles O(h^2k)
A=ones(k,k);
for i=1:k-1
    for j=1:k-1
      A(i+1,j+1)=(1/2^(2*i))^(j);
    end
end
A
B=zeros(k,1);
B(1,1)=1;
C=A\B;
C=C.';
```

Ekstrapoliacinis metodas:

<...>

```
mn = input('Norima tikslumo eile/2 = ');
if mn < 1
    mn=1;
end
m=1;
koef1 = koeficientai(2*m+1, 1, 0); % f^(1) skirt santykio central koef
koef2 = koeficientai(2*m+1, 2, 0); % f^(2) skirt santykio central koef
if m>1 % tuomet reikia forward ir backward koef.
  koeff1 = koeficientai(2*m+1, 1, 1); % f^(1) skirt santykio forward koef
  koefb1 = koeficientai(2*m+1, 1, -1); % f^(1) skirt santykio backward koef
  koeff2 = koeficientai(2*m+2, 2, 1); % f^(2) skirt santykio forward koef
  koefb2 = koeficientai(2*m+2, 2, -1); % f^(2) skirt santykio forward koef
end
koefextra2 = koeficientaiExtra2(mn).';
nlevels = input(' Iteraciju skaicius = ');

for k = 1:nlevels % grafikui paklaidu priklausomybei nuo pradinio zingsnio ekstrapoliacijoj
    n0 = 2*2^k + 21          %  pradinis (reikiamas apskaiciuoti norimu tikslumu)tinklelio
tasku skaicius
    h0=(xf-x0)/(n0-1)        % pradinis zingsnis, su kuriuo skaiciuojama m=1 tikslumo aprox,
su h/2 skaiciuojama 2k aukstesnes eiles aprox
% lygciu sistemos sudarymas
Yextra=zeros(n0,1);
dd=1;  n=n0;  h=h0;
for i = 1:n
  xo(i)=x0+(i-1)*h;
end
  Ytikras = sprendinys(xo).';
for mm=1:mn % ekstrapoliacija per mn zingsniu
if mm>1, n=(n-1)*2+1; dd=dd*2; h=h/2;end % zingsnio daliklis: 1, 2, 4, ... extrapoliacijos
elementams
clear x
 for i = 1:n
  x(i)=x0+(i-1)*h;
 end
 Ytikras2 = sprendinys(x).';
A=zeros(n,n); B=zeros(n,1); B(1)=y0;
B(n)=yf; A(1,1)=1; A(n,n)=1;
c=m+1; % centrinio elemento pozicija koef vektoriuje
% centrinis antrai ir desininis pirmai
iki= n-m; dum=2*m; ilgis=2*m+1; ilgis2=ilgis+1;
% kur centriniai skirt santykiai jau gali buti taikomi aproksimacijai, t.y.
% pakanka elementu i prieki ir atgal:
for i=c:iki
    A(i,i)  = a1(x(i))*koef2(1,c)/h/h+a2(x(i))*koef1(1,c)/h+a3(x(i)); % centrinis elementas
m+1
    B(i)=-a4(x(i));
    for ii=1:m
      A(i,i-ii)= a1(x(i))*koef2(1,c-ii)/h/h+a2(x(i))*koef1(1,c-ii)/h;
      A(i,i+ii)= a1(x(i))*koef2(1,c+ii)/h/h+a2(x(i))*koef1(1,c+ii)/h;
    end
end

A=sparse(A);
Y=A\B;
Y
if mm==1
    Yextra=koefextra2(mm)*Y;
    koefextra2(mm)
end
if mm>1
    koefextra2(mm)
    Y=koefextra2(mm)*Y;
    ind=1;
    for ii=1:n0
```

```
        if ii>1 ind=ind+dd; end
        Yextra(ii)=Yextra(ii)+Y(ind);
    end
end
length(Y)
length(Ytikras2)
 Eh = norm(Y - Ytikras2,inf);
 end
 Ehextr = norm(Yextra - Ytikras,inf);
 Ehvec(k)=Ehextr;
 hvec(k)=h0;
      figure(1)
      plot( xo,Ytikras,'*-',xo, Yextra,'o-');
      xlabel('x')
      ylabel('u(x)')
      title_str = ['zingsnis h = ', num2str(h0),'tikslumas O(h^2*',num2str(mn)];
      title(title_str);
      legend('Tikslus diskretus spr.','FD aproksimacija')
hold on;
set(gcf,'Color',[1 1 1]);
ezplot(sprendinys, [x0 xf]);
hold off;
disp('any key');
pause
end
  figure(2)
    loglog(hvec,Ehvec,'o', hvec,Ehvec)
    xlabel('h')
    ylabel('E_h = max |Y_{FD} - Y_{tikslus}|')
    title('Log-log grafikas paklaidos E_h keiciant zingsni h, esant ekstrapoliacija gautam
tikslumui 2*m')
    set(gcf,'Color',[1 1 1]);
%  Konvergavimo greitis
  conv_rate = mean( diff(log(Ehvec)) ./ diff(log(hvec)) );
  fprintf('Gautas konvergavimo greicio ivertis h^p, kur p = %4.3e\n',conv_rate);
```

3 priedas. Kai kurios 2.3.3 skyriaus programos

```
clear all
m=5;
Nr=3; % uzdavinio nr, zr. switch-case
%     -u_xx - u_yy = f(x,y), 0<x,y<1,
switch Nr
case 1
%-----------------------------------------------------------------
% 1.
sprendinys = @(x,y) sin(2*pi*x).*y.^2.*(1-y).^2;
f= @(x,y) 4*pi^2*sin(2*pi*x).*y.^2.*(1-y).^2 - sin(2*pi*x).*(2-12*y+12*y.^2);
maxx=1;
maxy=1;
%krastines reiksmes:
ba = @(x) sprendinys(x,0) ; % apatinis krastas y=0, x=0:maxx
bv = @(x) sprendinys(x, maxy); % virsutinis krastas y=maxy, x=0:maxx
bk = @(y) sprendinys(0, y);% kairysis krastas x=0, y=0:maxy
bd = @(y) sprendinys(maxx, y);% desinysis krastas x=maxx, y=0:mxy
%-----------------------------------------------------------------
case 2
%-----------------------------------------------------------------
maxx=1;
maxy=1;
f = @(x,y) -6*abs(x-sqrt(1/3)).*cos(2*pi*y.^2)-abs(x-sqrt(1/2)).^3.*(-
16*pi^2*y.^2.*cos(2*pi*y.^2)-4*pi*sin(2*pi*y.^2));
sprendinys=@(x,y) abs(x-sqrt(1/3)).^3*cos(2*pi*y.^2);
ba = @(x) sprendinys(x,0) ; % apatinis krastas y=0, x=0:maxx
```

```
bv = @(x) sprendinys(x, maxy); % virsutinis krastas y=maxy, x=0:maxx
bk = @(y) sprendinys(0, y);% kairysis krastas x=0, y=0:maxy
bd = @(y) sprendinys(maxx, y);% desinysis krastas x=maxx, y=0:mxy
%-----------------------------------------------------------------
case 3
%-----------------------------------------------------------------
maxx=1;
maxy=1;
 f =inline('4*pi*sin(2*pi*x)*( pi*cos(2*pi*y^2)*(1+4*y^2)+sin(2*pi*y^2) )');
 sprendinys=@(x,y)sin(2*pi*x).*cos(2*pi*y.^2);
%krastines reiksmes:
ba = @(x) sprendinys(x,0) ; % apatinis krastas y=0, x=0:maxx
bv = @(x) sprendinys(x, maxy); % virsutinis krastas y=maxy, x=0:maxx
bk = @(y) sprendinys(0, y);% kairysis krastas x=0, y=0:maxy
bd = @(y) sprendinys(maxx, y);% desinysis krastas x=maxx, y=0:mxy
end
nlevels = input(' Iteraciju skaicius = '); % max 10
if nlevels>10, nlevels=10; end
for kk = 1:nlevels
   % nx=20;
    nx = 1+3*m+kk*10;         %  Tinklelio tasku skaicius, programa pavilks iki mazdaug 100
    ny=nx;
    dx = maxx/(nx-1)
    x=linspace(0,maxx,nx);
    dy = maxy/(ny-1) ;
    y=linspace(0,maxy,ny);
if m < 1
    m=1;
end
koef2 = koeficientai(2*m+1, 2, 0); % f^(2) skirt santykio central koef
if m>1 % tuomet reikia forward ir backward koef.
  koeff2 = koeficientai(2*m+2, 2, 1); % f^(2) skirt santykio forward koef
  koefb2 = koeficientai(2*m+2, 2, -1); % f^(2) skirt santykio backward koef
end
nxny=nx*ny;
A=zeros(nxny,nxny);
B=zeros(nxny,1);

%-------------------------------------------------------------------------
% laisvas stulpelis
k=nx;
for j=2:ny-1
    for i=2:nx-1
        k=k+1;
        if i==2  k=k+1; end
        B(k,1)=-f(x(i),y(j));
        if i==nx-1, k=k+1; end
    end
end

% Krastines salygos:
% ------------------------------------------------------------------------

% krastines salygos staciakampes srities kairej ir desinej:
k=0;
for i = 1:nx:nxny
    k=k+1;
    B(i)=bk(y(k));
end
k=0;
for i = nx:nx:nxny
    k=k+1;
    B(i)=bd(y(k));
end
% krastines salygos staciakampes srities apacioj ir virsuj:
```

```
for i = 2:1:nx-1
   B(i)=ba(x(i));
end
k=1;
for i = nxny-nx+2:1:nxny-1
    k=k+1;
B(i)=bv(x(k));
end
% krastines salygos staciakampes srities kairej ir desinej:

for i = 1:nx:nxny
    A(i,i)= 1;
end
for i = nx:nx:nxny
    A(i,i)=1;
end
% krastines salygos staciakampes srities apacioj ir virsuj:
for i = 2:1:nx-1
A(i,i)=1;
end
for i = nxny-nx+2:1:nxny-1
A(i,i)=1;
end
%1. Sudarom forward, central, backward isilgai x asies, t.y. pagal gardeles
%eilutes, gausim matrica - busimos dideles matricos blokeli istrizainini
Ap=zeros(nx,nx);
c=m+1; % centrinio elemento pozicija koef vektoriuje
% centrinis antrai ir desininis pirmai
iki= nx-m;
dum=2*m;
ilgis=2*m+1;
ilgis2=ilgis+1;
% kai krastuose nepakanka tasku centriniam skirt, naudojam forward prdzioj,
% per 2m+2 taskus f^(2)
for i=2:m
    Ap(i,i)  = koeff2(1); % diagonalinis elementas
    for ii=1:ilgis
      Ap(i,i+ii)= koeff2(1+ii);
    end
end
% kur centriniai skirt santykiai jau gali buti taikomi aproksimacijai, t.y.
% pakanka elementu i prieki ir atgal:
for i=c:iki
    Ap(i,i)  = koef2(c); % centrinis elementas m+1
    for ii=1:m
      Ap(i,i-ii)= koef2(c-ii);
      Ap(i,i+ii)= koef2(c+ii);
    end
end
% ir backward gale:
for i=iki+1:nx-1
    Ap(i,i)  = koefb2(1,ilgis2); % diagonalinis elementas
    for ii=1:ilgis
      Ap(i,i-ii)= koefb2(ilgis2-ii);
    end
end
Ap=Ap/dx/dx;
% pridedame gauta diagonalini elementa (matrica Ap) prie pagrindines
% matricos A diagonales
for j=nx+1:nx:nxny-nx
    A(j:j+nx-1,j:j+nx-1)=A(j:j+nx-1,j:j+nx-1)+Ap;
end
dy2=dy^2;
% Gavome A, kur FD aproksimacija atlikta tik u_xx, dabar pridesime u_yy
```

```
% j - per gardeles eilutes, kuriose reikia forward, i - per visus gardeles tskus
% isilgai x (isskyrus krasta), kraste padidinam k, kad k butu matricos A
% eilutes nr
iki= ny-m;
k=nx; % pildomos A eilutes nr, atitinkantis omega_k gardeles taska,
      % cia k prasideda nuo 2-os gardeles eilutes, nes pirma jau uzpildyta krastinese
salygose.
for j=2:m
    for i=2:nx-1
        k=k+1;
         if i==2, k=k+1; end
    A(k,k)  =A(k,k)+ koeff2(1)/dy2; % diagonalinis elementas
    for ii=1:ilgis
      A(k,k+nx*ii)=A(k,k+nx*ii)+ koeff2(1+ii)/dy2;
    end
            if i==nx-1, k=k+1; end
    end
end
% kur centriniai skirt santykiai jau gali buti taikomi aproksimacijai, t.y.
% pakanka elementu i prieki ir atgal:
for j=c:iki
    for i=2:nx-1
      k=k+1;
       if i==2, k=k+1; end
      A(k,k)  = A(k,k)+koef2(c)/dy2; % centrinis elementas m+1
    for ii=1:m
      A(k,k-nx*ii)= A(k,k-nx*ii)+ koef2(c-ii)/dy2;
      A(k,k+nx*ii)= A(k,k+nx*ii)+ koef2(c+ii)/dy2;
    end
     if i==nx-1, k=k+1; end
    end
end
% ir backward gale:
for j=iki+1:ny-1
    for i=2:nx-1
      k=k+1;
      if i==2 , k=k+1; end
    A(k,k)  =A(k,k)+ koefb2(ilgis2)/dy2; % diagonalinis elementas
    for ii=1:ilgis
      A(k,k-nx*ii)=A(k,k-nx*ii)+ koefb2(ilgis2-ii)/dy2;
    end
     if  i==nx-1, k=k+1; end
    end
end
A=sparse(A);
figure(1)
spy(A~=0); % vizualizuojamas matricos elementu issidestymas
set(gcf,'Color',[1 1 1]);
  ufd = A\B;
  paklaida_sistemos=norm(A*ufd-B)
% sprendini is vektoriaus Ufd perrasom i matrica, kurioje bus gardeles
% taskus atitinkantys sprendiniu iverciai
Utikslus = zeros(nxny,1);
k=0;
for j=1:ny
    for i=1:nx
        k=k+1;
        Utikslus(k,1)=sprendinys(x(i),y(j));
    end
end
    Ufd = reshape(ufd,nx,ny);
    Utikslus=reshape(Utikslus,nx,ny);
    Eh = norm(Ufd(:)-Utikslus(:),'inf')
    %Eh = norm(Ufd(:)-Utikslus(:))
    Ehvec(kk) = Eh;
```

```
    hvec(kk) = dx;
    figure(2)
      subplot(321)
        mesh(x,y,Utikslus)
        xlabel('x')
        ylabel('y')
        title('Tikslus spr.')
      subplot(322)
        imagesc(Utikslus)
        colormap(hsv), colorbar
        xlabel('x')
        ylabel('y')
        title('Tikslus spr.')
      subplot(323)
        mesh(x,y,Ufd)
        xlabel('x')
        ylabel('y')
        title('FD aproksimacija')
      subplot(324)
        imagesc(Ufd)
        colormap(hsv),
        colorbar
        xlabel('x')
        ylabel('y')
        title('FD aproksimacija')
        subplot(325)
        mesh(x,y,Ufd-Utikslus)
        xlabel('x')
        ylabel('y')
        title('paklaida')
      subplot(326)
        imagesc(Ufd-Utikslus)
        colormap(hsv),
        colorbar
        xlabel('x')
        ylabel('y')
        title('paklaida')
      set(gcf,'Color',[1 1 1]);
    disp('any key to continue')
    pause
 end
  figure(3)
    loglog(hvec,Ehvec,'o', hvec,Ehvec)
    xlabel('dx')
    ylabel('Err =norm(u_fd - u_{tikslus})')
    title('Log-log grafikas paklaida-zingsnis')
    set(gcf,'Color',[1 1 1]);
  conv_rate = mean( diff(log(Ehvec)) ./ diff(log(hvec)) );
  fprintf('Konv. greicio ivertis h^p, cia p = %4.3e\n',conv_rate);
```

4 priedas. Kai kurios 2.3.4 skyriaus programos

```
function C=koeficientaiExtra(k)
% randami koeficientai extrapoliavimui, k - koeficientu vektoriaus ilgis,
% tuo paciu ir norima tikslumo eile O(h^k)
A=ones(k,k);
for i=1:k-1
    for j=1:k-1
      A(i+1,j+1)=(1/2^i)^(j);
    end
end
```

```
B=zeros(k,1);
B(1,1)=1;
C=A\B;
C=C.';

function [U] = daugiazBTCS(dt,t0,tmax,dx,nx,m,p,alpha,x,x0,xf, U, nuo) % O(h*2m)+O(t^p)
funkcija perduot ir nekartot uzdavinio
% p - eile per kiek sluoksniu t atzvilgiu skaiciuojama, p=1 - vienazingsnis
% --- daugiazingsnis metodas backward time central space
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L=xf-x0;
nt=ceil((tmax-t0)/dt +1);

% kai tikslumas O(h^2m), tai central approx reikia 2m+1 tasku, taigi koef
% vektorius bus 2m+1 ilgio
koef2 = koeficientai(2*m+1, 2, 0); % f^(2) skirt santykio central koef
if m>1 % tuomet reikia forward ir backward koef.
  koeff2 = koeficientai(2*m+2, 2, 1); % f^(2) skirt santykio forward koef
  koefb2 = koeficientai(2*m+2, 2, -1); % f^(2) skirt santykio backward koef
end
koefb1 = koeficientai(p+1, 1, -1) ;% f^(2) skirt santykio backward koef
c=m+1; % centrinio elemento pozicija koef vektoriuje
% % O(h), centrinis antrai ir desininis pirmai
iki= nx-m;
ilgis=2*m+1;
ilgis2=ilgis+1;
r = alpha*dt/dx^2;
ik=nuo+nt-2;
for k=nuo:ik % nuo+nt
A=zeros(nx,nx);
B=zeros(nx,1);
B(1)=0;
B(nx)=0;
  % kai krastuose nepakanka tasku centriniam skirt, naudojam forward prdzioj,
% per 2m+2 taskus f^(2)
for i=2:m
    A(i,i)  = koeff2(1); % diagonalinis elementas

    for ii=1:ilgis
      A(i,i+ii)= koeff2(1+ii);
    end
end

% kur centriniai skirt santykiai jau gali buti taikomi aproksimacijai, t.y.
% pakanka elementu i prieki ir atgal:
for i=c:iki
     A(i,i) = koef2(c); % diagonalinis elementas
    for ii=1:m
      A(i,i-ii)=koef2(c-ii);
      A(i,i+ii)=koef2(c+ii);
    end
end
% ir backward gale:
for i=iki+1:nx-1
    A(i,i)  = koefb2(ilgis2);
    for ii=1:ilgis
      A(i,i-ii)=koefb2(ilgis2-ii);
    end
end

for pp=1:p
B(2:nx-1)=B(2:nx-1)+koefb1(p-pp+1)*U(2:nx-1,k-pp); %p-pp+1 k-pp
end
A=r*A;
A(1,1)=1;
```

```
A(nx,nx)=1;
for j= 2:nx-1
    A(j,j)=A(j,j)-koefb1(p+1);
end


A=sparse(A);
U(:,k)=A\B;
%errsist=norm(A*U(:,k)-B(:))
%U(nx,k)=0;
end
% grazina matrica U(:,t0:dt:tmax)
function [err,x,t,U] = heatBTCSextra(dt,dx,tmax,m,p) % schema su extrapoliacija
inicializavimui
clc
if nargin<2, dx = 0.25; end
if nargin<3, tmax = 20; end
if nargin<4, m = 2;    end %O(h^2m)
if nargin<5, p=4;      end %O(t^p)
if nargin<1, dt = dx;  end

 % tikslumo eile m=1 O(h^2), m=2 O(h^4) ir t.t.

Nr=1; % uzdavinio nr, zr. switch-case
% sprendziamas nestacionarus silumos laidumo uzdavinys:
% du/dt= alpha*d^2u/dx^2 +f(x,t)

xf=4; x0=0;
L=xf-x0;
nx=ceil(L/dx +1);
nt=ceil(tmax/dt +1);

if 3*m+1> nx
    m=floor((nx-1)/3);
end


x = linspace(0,L,nx)';
% imsim t0=0;
t = linspace(0,tmax,nt);
U = zeros(nx,nt);
switch Nr
case 1
%-----------------------------------------------------------------
% 1.

alpha = 0.2;
% krastines reiksmes
sprendinys = @(x,t) sin(pi*x/L).*exp(-t*alpha*(pi/L)^2); % tikslus dif. lygties sprendinys
U(:,1) = sprendinys(x,0);% pradine salyga t=0
U(1,1)=0; U(nx,1)=0;
u0=0; % krastines salygos
uL=0;
%-----------------------------------------------------------------
end
m;

%%%%%%%%%%%%%Inicializacija tikslumu O(t^2m)%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% --- daugiazingsnis metodas backward time central space

koefextra = koeficientaiExtra(p).';
%if pp>nt
```

```
for pp=2:p
    %if pp>=nt
    %pildom pp-taji busimos inicializacijos sluoksni
     %1*O(t), veliau +koef*O(t^2)+...
     U(:,pp) = zeros(nx,1);
    for i=1:p %skaiciuojam 1/2 zingsnio buvusio
        ni=i-1;
        ni2=2^ni;
        dtnaujas=dt/ni2;
        % ieskot klaidos cia:
        Un=daugiazBTCS(dtnaujas,dt*(pp-2),dt*(pp-1),dx,nx,m,1,alpha,x,x0, xf, U(:,1:pp-1),
pp);%(:,ni2); U(:.pp-1) - buves sluoksnis yra naujo pradine salyga
        %             (dt,     ,t0,       tmax,     dx,nx,m,p,alpha,x,x0,xf, U,        nuo)
        %
        %Un(:,size(Un,2));%pasiimam sprendini paskutiniu laiko momentu, jis
        %apskaiciuotas tikslumu O(t^pp)
        U(:,pp) = U(:,pp)+koefextra(i).*Un(:,size(Un,2));
        err = norm(Un(:,size(Un,2))-sprendinys(x,(pp-1)*dt));
    end
   % disp('----------------------------------------')
    err = norm(U(:,pp)-sprendinys(x,(pp-1)*dt));
   % disp('----------------------------------------')
end
U;
% gavome pradines salygas - p sluoksniu daugiazingsniam metodui pradeti
% toliau  U(:,1:p) naudojami daugiazingsnio metodo inicializacijai
U=daugiazBTCS(dt,0,tmax,dx,nx,m,p,alpha,x,x0,xf,U(:,1:p),p+1);
% paklaidos
U;
ue=sprendinys(x,tmax);
sprendinys(x(nx),tmax);
%ue(nx)=0;
%ue(nx)=uL;
U(:,nt)-ue;
err = norm(U(:,nt)-ue);


function [err,x,t,U] = heatBTCS_extrapol_vienazingsnis(dt,dx,tmax,m,p) % tikslumui pasiekti -
vien ekstrapoliacija, schema O(t)
clc
if nargin<2, dx = 0.25; end
if nargin<1, dt = dx;  end
if nargin<3, tmax = 20; end
if nargin<4, m = 4;    end %O(h^2m)
if nargin<5, p=2;      end %O(t^p)
 % tikslumo eile m=1 O(h^2), m=2 O(h^4) ir t.t.
Nr=1; % uzdavinio nr, zr. switch-case
% sprendziamas nestacionarus silumos laidumo uzdavinys:
% du/dt= alpha*d^2u/dx^2 +f(x,t)

xf=4; x0=0;
L=xf-x0;
nx=ceil(L/dx +1);
nt=ceil(tmax/dt +1);

if 3*m+1> nx
    m=floor((nx-1)/3);
end


x = linspace(0,L,nx)';
% imsim t0=0;
t = linspace(0,tmax,nt);
U = zeros(nx,nt);
switch Nr
```

```
case 1
%-------------------------------------------------------------
% 1.
f=inline('0','x','t') ;
alpha = 0.2;
% krastines reiksmes
sprendinys = @(x,t) sin(pi*x/L).*exp(-t*alpha*(pi/L)^2); % tikslus dif. lygties sprendinys
U(:,1) = sprendinys(x,0);% pradine salyga t=0
U(1,1)=0; U(nx,1)=0;
u0=0; % krastines salygos
uL=0;
%ue = sprendinys(x,tmax);
%-------------------------------------------------------------
end

%%%%%%%%%%%%%Inicializacija tikslumu O(t^2m)%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% --- daugiazingsnis metodas backward time central space
koefextra = koeficientaiExtra(p).';

for pp=2:nt
    %if pp>=nt
    %pildom pp-taji busimos inicializacijos sluoksni
     %1*O(t), veliau +koef*O(t^2)+...
     U(:,pp) = zeros(nx,1);
    for i=1:p % skaiciuojam 1/2 zingsnio buvusio
        ni=i-1;
        ni2=2^ni;
        dtnaujas=dt/ni2;
        % ieskot klaidos cia:
        Un=daugiazBTCS(dtnaujas,dt*(pp-2),dt*(pp-1),dx,nx,m,1,alpha,x,x0, xf, U(:,1:pp-1),
pp);%(:,ni2); U(:.pp-1) - buves sluoksnis yra naujo pradine salyga
        %             (dt,     ,t0,       tmax,     dx,nx,m,p,alpha,x,x0,xf, U,          nuo)
        %
        %Un(:,size(Un,2));%pasiimam sprendini paskutiniu laiko momentu, jis
        %apskaiciuotas tikslumu O(t^pp)
        U(:,pp) = U(:,pp)+koefextra(i).*Un(:,size(Un,2));
        err = norm(Un(:,size(Un,2))-sprendinys(x,(pp-1)*dt));
    end
    %disp('----------------------------------------')
    errextr = norm(U(:,pp)-sprendinys(x,(pp-1)*dt));
    %sk=U(:,pp)-U(:,pp-1)
   % U(:,pp);
   % disp('----------------------------------------')
end

U;
ue=sprendinys(x,tmax);
sprendinys(x(nx),tmax);
%ue(nx)=0;
%ue(nx)=uL;
U(:,nt)-ue;
err = norm(U(:,nt)-ue);

function XTkeiciant_m_ir_p(maxp, maxm) % rodo su kokiais parametru rinkiniais konverguoja
metodas konkreciam uzdaviniui konkretiems h it t
if nargin<1, maxp = 8; end
if nargin<1, maxm = 4; end
clc
dx=4/500;
dt=4/500;
tmax=4;
%[P,M] = meshgrid(1:1:maxp,1:1:maxm);
```

```
P=1:1:maxp;
M=1:1:maxm;
for p=1:maxp
   for m=1:maxm
%[err, x,t,U]=heatBTCS_extrapol_vienazingsnis(dt,dx,tmax,m,p);
[err, x,t,U]=heatBTCSextra(dt,dx,tmax,m,p);
if err>1
    err=1;
end
if isnan(err)
    err=-1;
end
ERR(p,m) = err;
   end
end
ERR
imagesc(M,P,ERR, [-1,1])
colormap(jet)
colorbar;
xlabel('m, kur O(h^{2m})')
ylabel('p, kur O(t^p)')
set(gcf,'Color',[1 1 1]);
```