KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

Vilma Staliulionytė

DAUGIAMAČIO TANKIO ĮVERTINIO, PAGRĮSTO TIKSLINIU PROJEKTAVIMU, TYRIMAS

Magistro darbas

Vadovas
dr. M. Kavaliauskas

KAUNAS, 2014

KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. dr. N. Listopadskis
2014 06 02

DAUGIAMAČIO TANKIO ĮVERTINIO, PAGRĮSTO TIKSLINIU PROJEKTAVIMU, TYRIMAS

Taikomosios matematikos magistro baigiamasis darbas

Vadovas
dr. M. Kavaliauskas
2014 06 02

Recenzentas
doc. dr. A. Kabašinskas
2014 06 02

Atliko
FMMM-2 gr. stud.
V. Staliulionytė
2014 05 30

KAUNAS, 2014

KVALIFIKACINĖ KOMISIJA

Pirmininkas:	Juozas Augutis, profesorius (VDU)
Sekretorius:	Eimutis Valakevičius, profesorius (KTU)
Nariai:	Arūnas Barauskas, direktoriaus pavaduotojas (UAB „Danet Baltic“)
	Vytautas Janilionis, docentas (KTU)
	Zenonas Navickas, profesorius (KTU)
	Kristina Šutienė, docentė (KTU)
	Jonas Valantinas, profesorius (KTU)

Staliulionytė V. Analysis of Projection pursuit density estimator: Master‘s work in applied mathematics / supervisor dr. M. Kavaliauskas; Department of Applied mathematics, Faculty of Mathematics and Natural Sciences, Kaunas University of Technology. Kaunas, 2014. – 55p.

Density distribution is one of the most important features characterizing a random variable, therefore its estimation is very important. In this paper nonparametric statistical estimation of multivariate probability density is presented. For that purpose, the model of projection pursuit density estimation (PPDE) was chosen. The projection pursuit density estimation idea was proposed by J. H. Friedman (1974). [30] The author suggests these PPDE parts:

1) Find all interesting univariate projections. The more projection’s univariate density differs from standard Gaussian density, the more interesting it is;
2) Apply a transformation to the data which removes the structure in the found projection direction, so that its projection has standard Gaussian density;
3) Estimate multivariate probability density of the data using the found interesting projection directions.

The main purpose of this work is multivariate probability density estimation using projection pursuit density estimation method.

The main tasks:

- Analysis of Projection pursuit density estimator with different data samples;
- Analysis of Projection pursuit density estimator using Mardia‘s skewness and kurtosis, Henze – Zirkler, Royston H tests for normality;

Findings and recommendations for approaches which produce the smallest errors for samples of small and large sizes are presented in the conclusions section.

Staliulionytė V. Daugiamačio tankio įvertinio, pagrįsto tiksliniu projektavimu, tyrimas: Taikomosios matematikos magistro baigiamasis darbas / vadovas dr. M. Kavaliauskas; Taikomosios matematikos katedra; Matematikos ir gamtos mokslų fakultetas, Kauno technologijos universitetas. Kaunas, 2014. - 55p.

Darbe nagrinėjamas neparametrinis tankio vertinimas tikslinio projektavimo metodu (angl. Projection Pursuit Density Estimation). Remiamasi J. H. Friedman 1974 metais pasiūlytu algoritmo sudarymo modeliu:

1. Ieškoma duomenų projekcijų skirstinio labiausiai nepanašaus į standartinį Gauso skirstinį;
2. Rastoje projektavimo kryptyje keičiama duomenų struktūra, duomenų skirstinys tampa standartinu Gauso skirstiniu, tokiu būdu išvengiama anksčiau surastos projekcijos aptikimo;
3. Vertinamas daugiamatis tankis naudojant surastas vienamates projekcijas.

Darbo tikslas ir uždaviniai – daugiamačio skirstinio tankio įvertinio, pagrįsto tiksliniu projektavimu, tyrimas. Daugiamačio skirstinio tankio įvertinio tikslumo analizė esant skirtingų tūrių duomenų imtims, taip pat naudojant domenų normalumo testus.

Atlikus daugiamačio skirstinio tankio įvertinio tikslumo analizę, išvadose pateikiami pastebėjimai bei rekomendacijos apie metodus, kuriais gaunamos mažiausios paklaidos, esant mažoms ir didelėms imtims.

Turinys

PAVEIKSLŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS

ĮVADAS

Skirstinio tankis – viena iš svarbiausių funkcijų apibūdinančių atsitiktinį dydį, todėl jo vertinimas itin svarbus. Duomenų analizėje žinoma daug neparametrinių tankio vertinimo metodų. Vienas iš jų - tikslinio projektavimo metodas daugiamačio tankio vertinimui (angl. Projection Pursuit Density Estimation (PPDE)) nagrinėjamas baigiamajame darbe.

Didėjant stebimo dydžio dimensijai, neparametrinio tankio vertinimo metodai, pavyzdžiui branduolinis tankio įvertinimas, susiduria su „daugiamatiškumo prakeikimu“ – reikalingas itin didelis duomenų kiekis tankio įverčiui rasti. Vienas iš būdų padedančių išvengti šios problemos – tikslinio projektavimo metodo panaudojimas daugiamačiam tankiui vertinti.

J. H. Friedman 1974 metais [18] pasiūlė tikslinio projektavimo algoritmo sudarymo idėją:

1) Randamos visos „įdomios“ vienamatės duomenų projekcijos. Jų skirstinio tankis kaip įmanoma labiau skiriasi nuo standartinio Gauso skirstinio;
2) Atliekama duomenų transformacija surastoje projektavimo kryptyje pakeičianti duomenų skirstinio tankį į standartinį normalųjį;
3) Atliekama daugiamačio tankio vertinimo procedūra naudojant surastas projekcijas.

Pagrindinis darbo tikslas – įvertinti daugiamačio skirstinio tankį tikslinio projektavimo metodu.

Darbo uždaviniai:

- Ištirti daugiamačio skirstinio tankio įvertinio tikslumą esant įvairiems imties tūriams, duomenų dimensijai, klasių persidengimo lygiui, naudojamų Ležandro polinomų eilei; taip pat naudojant Mardia asimetrijos ir eksceso, Henze – Zirkler, Royston H normalumo testus.
- Apibendrinus tyrimo rezultatus pasiūlyti metodą optimaliam projektavimo krypčių kiekio parinkimui.

Darbą sudaro įvadas, bendroji ir tiriamoji dalys, išvados, literatūros sąrašas bei priedai. Bendrojoje dalyje aptariamas tikslinio projektavimo algoritmo sudarymas, duomenų generavimas, paklaidų matavimo metrikų bei testų duomenų normalumui tikrinti pasirinkimas. Tiriamojoje dalyje grafiškai iliustruojami mišinių skirstinių persidengimo lygmenys, pateikiama tikslinio projektavimo algoritmo iliustracija R^2 atveju; atliekamas tankio vertinimas renkantis skirtingus imties tūrius, duomenų normalumo testus.

2014 metais darbo tema buvo perskaitytas pranešimas KTU konferencijoje „MATEMATIKA IR MATEMATIKOS DĖSTYMAS – 2014“. Taip pat 2014 metais KTU XII -ojoje studentų konferencijoje „Taikomoji matematika“ buvo pristatytas pranešimas bei paruoštas straipsnis „Tankio vertinimo paklaidos augimo tyrimas naudojant perteklinį projektavimo krypčių kiekį“.[23] 2013 metais KTU XI-ojoje studentų konferencijoje „Taikomoji matematika“ buvo perskaitytas pranešimas ir paruoštas straipsnis tema „Tikslinio projektavimo panaudojimas duomenų vizualizavimui“. [24]

1. BENDROJI DALIS

1.1. TIKSLINIO PROJEKTAVIMO ALGORITMO APŽVALGA

Dažniausiai, ypač tyrimo pradžioje atliekama duomenų žvalgomoji analizė, t.y. surenkama informacija reikalinga tolimesniems tyrimams. Tokiu būdu gaunamos žinios bei įgyjamas supratimas apie nagrinėjamą reiškinį.

Aprašomojoje statistikoje naudojamos statistikos suvestinės siekiant apibendrinti stebėjimų rinkinį ir gauti kuo daugiau informacijos. Daugiamačių duomenų analizėje statistikos suvestinė remiasi vietos ir skalės matavimo kintamaisiais ir jų koreliacine struktūra. Klasikinė daugiamatė statistinė analizė suteikia galingus įrankius minėtoms charakteristikoms rasti. Jei duomenų išsidėstymas yra panašus į elipsiškai simetrinį pasiskirstymą, pavyzdžiu gali būti normalusis Gauso skirstinys, daugiamačių kintamųjų erdvėje, tuomet ši statistikos suvestinė dažniausiai suteikia daug svarbios informacijos.

Kai kuriais atvejais duomenų struktūra nėra pilnai nusakoma tiesiniais ryšiais (koreliacijomis) egzistuojančiais tarp kintamųjų. Be tiesinių ryšių, egzistuoja ir netiesinė statistinio ryšio forma kvadratinė, logaritminė, eksponentinė ir t.t. [10]

Kadangi pagrindinė funkcija, apibūdinanti atsitiktinį dydį yra tankio funkcija, todėl labai svarbu tiksliai jį įvertinti.

Nagrinėjant daugiamačius duomenis, tyrėjai dažniausiai ieško duomenų poerdvių, kurie būtų „įdomūs“, t.y. ieškoma tokių duomenų struktūrų, kad jų skirstinys turėtų Gauso skirstinio tankio pavidalą. Tačiau įvertinti daugiamačių duomenų rinkinius bei atlikti daugiamačio tankio vertinimą, esant didelei stebimo dydžio dimensijai, dažnai yra gana sudėtinga, vien todėl, kad didelių matavimų erdvėje, turint nedidelės imties duomenis, jie išsidėsto labai retai, vienas nuo kito dideliais atstumais. Todėl gana dažnai susiduriama su „daugiamatiškumo prakeikimo“ problema (angl. curse of dimensionality), t.y. mažai imties taškų patenka į didelio spindulio aplinką. Tai reiškia, kad reikalingas itin didelis stebėjimų kiekis tankį vertinant klasikiniais metodais, tokiais kaip – branduolinis tankio įvertinimas ar artimiausių kaimynų metodas. [19]. „Daugiamatiškumo prakeikimo“ problemos sprendimą savo darbe nagrinėja P. J. Huber [13]. Vienas iš jų - tikslinio projektavimo algoritmo taikymas vertinant daugiamačių duomenų pasiskirstymo tankį.

Nagrinėjant daugiamačių duomenų rinkinius, siekiama sumažinti jų dimensijas naudojant projektavimo strategijas, jų aptarimas pateikiamas S. L. Crawford ir T. C. Fall darbe [19]. Kaip yra žinoma, žmogaus gebėjimas atpažinti yra ribotas, geriausių rezultatų pasiekiama dirbant su mažomis dimensijomis. Įprasti vaizdavimai atliekami dvimatėje matavimų erdvėje Tačiau yra būdų kaip peržiūrėti ir didesnių dimensijų vaizdus. Tam pasitarnauja spalvų naudojimas kompiuterinėje

grafikoje, jis suteikia galimybes peržiūrėti trimačius, keturmačius, o taip pat ir dar didesnių dimensijų vaizdus. Taigi grafiškai tiriant duomenis didesnėse dimensijose, ieškoma juos atskleidžiančių žemesnių dimensijų vaizdų. Apie duomenų projekcijų suradimą labai plačiai aprašyta J. H. Friedman veikale [16].

Šiam tikslui įgyvendinti yra naudojami paprasti ir gana lengvai interpretuojami dimensijų mažinimo metodai – projekcijos metodai, jais daugiamačiai duomenys atvaizduojami žemesnės dimensijos erdvėje. Projekcijos metodų „tikslas – pateikti daugiamačius duomenis mažesnės dimensijos erdvėje taip, kad būtų kiek galima tiksliau išsaugota analizuojamų duomenų struktūra.“ [11]

Dažniausiai daugiamačių duomenų struktūriškumas yra matomas žemesnio matavimo dimensijose ir kiekvienos iš jų peržiūra gali suteikti papildomos informacijos. Taigi yra labai svarbu, kad tikslinio projektavimo algoritmas rastų kaip galima daugiau tokių vaizdavimų.

Praktikoje dažniausiai naudojami tiesiniai projekcijos metodai, ieškantys tiesinio poerdvio - plokštumos arba tiesės. [11] Šiais metodais yra „ieškoma tiesinės analizuojamų duomenų transformacijos“ galimybių: pasukimo, postūmio, atspindžio, suspaudimo ir pakeitimų. [12]

Projekcijos suradimo arba kitaip dar vadinamo tikslinio projektavimo algoritmo idėja – rasti daugiamačių duomenų vienamates projekcijas, kurios atskleistų informaciją apie duomenų rinkinio įgytą struktūrą. Išskiriami, rankinis ir automatizuotas metodai, kurių buvo imtasi siekiant atlikti tikslinį projektavimą.

Duomenų projekcijų ieškojimui rankiniu būdu naudojamos sklaidos diagramos. Vienas iš būdų informatyviam duomenų atvaizdavimui - braižyti kintamųjų taškų sklaidos diagramą, leidžiančią įvertinti statistinio ryšio tendenciją ir formą. Analizuojant tokiu būdu gautą vaizdinę medžiagą, galima aptikti įvairias struktūras, kokių nebuvo tikimasi rasti. Tai paprasčiausia vaizdavimo forma - vienu metu pateikiama visų duomenų projekcijų porų dvimatis vaizdas. Tačiau toks duomenų vaizdavimas leidžia aptikti struktūrą tik tarp atskirų porų projekcijų. Todėl esant didelei duomenų dimensijai reikia ieškoti kitų projektavimo metodų, jie aptariami [19].

Pats didžiausias rankinio tikslinio projektavimo metodo trūkumas – toks duomenų projekcijų išsamus vertinimas užima itin daug laiko. Jeigu kas nors pabandytų realizuoti Asimov Grand Tour procedūrą [20], atvaizduojančią nežymiai besiskiriančius duomenų projekcijų rinkinius vieną po kito – rezultatų pateikimas yra panašus į filmuko žiūrėjimą, siekiant ištirti daugiamačių duomenų erdvę. Tokia procedūra tiriant keturmačių duomenų erdvę užtrunkama daugmaž tris valandas. Plačiau apie šį metodą galima rasti [13].

Taigi tiriant daugiamačius duomenis, siekiant nustatyti jų išsidėstymą daugiamatėje erdvėje 1974 metais J. H. Friedman ir J. Tukey [18] pristatė algoritmą automatizuojantį tikslinio projektavimo metodą. Pagrindinė algoritmo idėja buvo priskirti kiekvienai projekcijai skaitinį indeksą, nusakantį tos

projekcijos struktūriškumą ir atlikti projektavimo indekso maksimizaciją jį sudarančių parametrų atžvilgiu. Įvairūs paieškos metodai tiksliniam projektavimui taip pat yra siūlomi ir P. A. Tukey, J. W. Tukey darbe. [21]

Tikslinio projektavimo principas remiasi surastos „įdomios“ projekcijos pašalinimu iš duomenų imties. Procesas tęsiamas tol, kol nebelieka „įdomių“ projekcijų – šių projekcijų skirstinys turi būti labai besiskiriantis nuo Gauso skirstinio. Būdų kaip pašalinti duomenų struktūrą pateikiama [13] ir [16] literatūros šaltiniuose.

Tikslinio projektavimo algoritmo sudarymas yra didelis žingsnis sprendžiant problemas susijusias su daugiamačiais duomenimis. [22]

1.2. TIKSLINIO PROJEKTAVIMO ALGORITMAS IR JO ANALIZĖ

1.2.1. PRADINIŲ DUOMENŲ SFERINIMAS

Siekiant atlikti turimų duomenų analizę naudojantis tikslinio projektavimo algoritmu, pirmiausia yra atliekami skaičiavimo trukmės sutaupymai, t.y. atliekamas pradinių duomenų sferinimas. Šiai procedūrai įgyvendinti pasitelkiama tiesinė transformacija – atliekami duomenų pasukimo, vietos ir mastelio pakeitimai.

Tarkime, kad Y yra atsitiktinis kintamasis erdvėje R^p. Atliekame kovariacinės matricos Σ skaidymą pagal tikrines reikšmes ir tikrinius vektorius:

$$\Sigma = E[(Y - EY)(Y - EY)^T] = UDU^T \qquad \textbf{(1.1)}$$

Šioje išraiškoje U – ortonormuota matrica, matricos Σ tikrinių vektorių matrica, kurios stulpeliai užrašyti vektoriais yra vienetinio ilgio ir ortogonalūs, taip pat tokios matricos gali būti vadinamos ir ortogonaliomis; D diagonalinė $p \times p$ matrica. Tuomet susferintus kintamuosius gauname taip:

$$Z = \left(\frac{1}{\sqrt{D_j}}\right) U(Y - EY) \qquad \textbf{(1.2)}$$

Sakykime, kad q yra kovariacinės matricos Σ rangas, tuomet sferintų kintamųjų Z visos q komponentės yra gaunamos naudojantis **(1.3)** išraiška:

$$Z_j = \left(\frac{1}{\sqrt{D_j}}\right) \sum_{i=1}^{p} U_{ij}(Y_i - EY_i)\,, 1 \le j \le q \qquad \textbf{(1.3)}$$

Pagal prielaidą ortonormuotos matricos U ir diagonalinės matricos D eilutės yra surikiuojamos mažėjančia D_j reikšmių tvarka. Tuomet pagal apibrėžimą gaunama, kad sferintų kintamųjų Z vidurkis $E[Z] = 0$, o Z kovariacinė matrica $E[ZZ^T] = I$ - vienetinė matrica.

Po pradinėje analizės stadijoje atlikto duomenų sferinimo gaunami akivaizdūs skaičiavimų trukmės sutaupymai.

Bet kurios tiesinės kombinacijos

$$X = \alpha^T Z = \sum_{i=1}^{q} \alpha_i Z_i \qquad \textbf{(1.4)}$$

su dispersija

$$var(X) = \alpha^T \alpha = \sum_{i=1}^{q} \alpha_i^2 \qquad \textbf{(1.5)}$$

esant apribojimui $\alpha^T \alpha = 1$ užtikrina, kad kiekviena iš tiesinių kombinacijų aprašytų **(1.4)** išraiškoje turi vienetinę dispersiją.

Taip pat reikia paminėti, kad dvi tiesinės kombinacijos, sudarytos naudojant ortogonalius vektorius yra nekoreliuotos. Tai yra, $\alpha^T \beta = 0$ reiškia, kad

$$E[(\alpha^T Z)(\beta^T Z)] = 0 \qquad \textbf{(1.6)}$$

Visa tolimesnė tikslinio projektavimo algoritmo teorinė analizė, o ir tiriamojoje dalyje atliekami praktiniai skaičiavimai remiasi susferintų kintamųjų panaudojimu. Reikia pabrėžti, kad tik skaičiavimų pabaigoje gautąjį sprendinį transformuojame atgal į originalias pradines Y koordinates.

Kadangi duomenų sferinimas atliekamas vieną kartą – pačioje skaičiavimų pradžioje, taip gaunama naudos – atliekamas skaičiavimų trukmės sumažinimas.

Pagal apibrėžimą visi sferinti Z kintamieji yra afiniškai invariantiški, todėl bet kuris projekcijos indeksas tai pat paveldės šią savybę, t.y. bus nekintantis postūmio, poslinkio ir mastelio atžvilgiu.

1.2.2. PROJEKTAVIMO INDEKSAS

Projektavimo indeksas – tai tikslinio projektavimo algoritmo šerdis. Šio algoritmo tikslas yra surasti daugiamačių duomenų „įdomiausias“ projekcijas, t.y. tokias projektavimo kryptis, kuriose duomenų skirstinio tankio kreivė labiausiai skiriasi nuo standartinio Gauso skirstinio tankio kreivės.

Iš pirmo žvilgsnio atrodo, kad pasirinktas toks įvertinimas gali būti sudėtingas, tačiau tam, kad šį užmojį paversti suprantamu yra naudojamas projektavimo indeksas, kuris yra projektuojamų duomenų skirstinio funkcionalas. Šis funkcionalas turi būti tolydus, įgauti didelę reikšmę, kai suprojektuotų duomenų skirstinys yra apibrėžiamas kaip „įdomus“, t.y. labiausiai besiskiriantis nuo standartinio Gauso skirstinio ir mažą reikšmę – priešingu atveju.

„Įdomumo“ sąvoka gali skirtis priklausomai nuo algoritmo taikymo, tai pastebima P. J. Huber straipsnyje [22]. Mūsų atveju reikia rasti papildomą struktūriškumą, kuris nėra užfiksuotas koreliacinėje duomenų struktūroje. Vienas iš būdų tai užtikrinti, t.y. padaryti, kad projekcijos indeksas būtų nekintamas afininių transformacijų, tokių kaip: poslinkiai, posūkiai, postūmiai, suspaudimai ar ištempimai, atžvilgiu daigiamatėje erdvėje.

„Įdomumo“ sąvoką gali būti sunku įvertinti kiekybiškai, tačiau „ne įdomumo“ sąvoką galima paaiškinti gana paprastai. P. J. Huber [13] ir M. C. Jones [14] pateikė tvirtus euristinius argumentus, kada normalusis skirstinys turi būti laikomas mažiausiai „įdomiu“:

- Daugiamačio normaliojo skirstinio tankis yra pilnai apibrėžtas savo tiesine struktūra (vieta ir kovariacine matrica).
- Visos daugiamačio normaliojo skirstinio projekcijos taip pat yra normalieji skirstiniai. Išsamiau šį teiginį galima paaiškinti taip: jei tikslinio projektavimo algoritmu rasta mažiausiai normali projekcija nereikšmingai skiriasi nuo projekcijos su normaliuoju skirstiniu, interpretuojama, taip kad, daugiamatis duomenų skirstinys yra normalusis.
- Jeigu egzistuoja tiesinė kombinacija, kur duomenys yra struktūrizuoti – išsiskiria į kelias klases, daugelis tiesinių kombinacijų turės skirstinį artimą standartiniam Gauso skirstiniui.
- Esant fiksuotai dispersijai, normalusis skirstinys turi mažiausiai informacijos. [15]

Mūsų tikslas yra netiesinio struktūriškumo suradimas, iš esmės daugiamačio normaliojo skirstinio elipsiškas simetriškumas yra tai, dėl ko jis yra laikomas mažiausiai „įdomiu“. Normaliojo skirstinio skaičiuojamumo savybė daro jį labiausiai tinkamu pasirinkimu iš elipsiškai simetrinių skirstinių šeimos.

Remiantis šia nuomone, bet kuri statistika duomenų normalumo tikrinimui gali būti projektavimo indekso pamatu. Turint skirtingus skirstinius, duomenų normalumo tikrinimo statistikų jautrumas gali būti įvairus. Dažniausiai normalumo testai skiria didelį dėmesį sunkioms skirstinio uodegoms. Mūsų tikslas – rasti kryptis kuriose suprojektavus daugiamačius duomenis būtų gaunami daugiamodaliai skirstiniai. Tokie skirstiniai nuo normaliojo Gauso skirstinio labiau skiriasi savo centrine dalimi, o ne uodegomis. Todėl yra ieškomas projektavimo indeksas, kuris pabrėžia nuokrypius nuo normalumo pagrindinėje skirstinio dalyje ir suteikia atitinkamai mažiau svorio uodegoms.

Kadangi projektavimo indeksas pasitarnauja kaip tikslo funkcija daugiaparametriniam optimizavimui, jo skaičiuojamumo savybės yra labai svarbios. Todėl reikalaujame, kad duotajam parametrų rinkiniui projektavimo indekso reikšmė būtų greitai skaičiuojama. Taip pat projektavimo indeksas turi būti tolydus, kad bent pirmosios išvestinės su turimomis parametrų reikšmėmis egzistuotų visur; todėl ir išvestinės taip pat turi būti greitai apskaičiuojamos.

Geriausių, skaičiuojamumo prasme, projektavimo indeksų sudarymas remiasi polinominiais momentais. Tam nėra reikalingas suprojektuotų reikšmių rikiavimas, o ir pačių polinomų reikšmės taip pat kaip ir jų išvestinės yra greit apskaičiuojamos pasitelkiant rekursines formules. Kadangi svarbu įvertinti nuokrypius nuo normalumo, viena iš galimybių - remtis projektavimo indeksu pagrįstu standartizuotais projektuojamo skirstinio kumuliantais, kaip siūlo P. J. Huber [26, 445psl.], pavyzdžiui Ermito polinomų momentai. Kita alternatyva pateikta M. C. Jones [14], jis siūlo naudoti projektavimo indeksą pagrįstą standartizuotų kumuliantų kvadratų sumomis.

Šio modelio skaičiuojamumo savybės tikrai atrodo puikiai, tačiau jis yra netinkamas, todėl kad, tokie projektavimo indeksai labai smarkiai pabrėžia nuokrypius nuo normalumo skirstinio uodegose. Pavyzdžiui, projektuojamas skirstinys su šiek tiek sunkesnėmis nei įprasta uodegomis, įgyja gan didelę projektavimo indekso reikšmę.

Prieš pradėdami teorinę tikslinio projektavimo algoritmo indekso analizę, apibrėžiame, kad dirbama su daugiamačiu tolydžiuoju tikimybiniu skirstiniu. Praktiniuose taikymuose tikėtinos reikšmės yra pakeičiamos atitinkamais aritmetiniais vidurkiais. Toliau pateikiamose tikslinio projektavimo algoritmo išraiškose didžiosios raidės reiškia atsitiktinius kintamuosius, o mažosios, dažniausiai su indeksais, bus naudojamos pažymėti šių dydžių imties stebėjimų reikšmes.

Vienos dimensijos tiksliniame projektavime ieškome tiesinės kombinacijos

$$X = \alpha^T Z \tag{1.7}$$

tokios, kad tikimybiniai tankiai $p_\alpha(X)$ būtų kaip įmanoma labiau struktūrizuoti – labiausiai besiskiriantys nuo standartinio Gauso skirstinio tankio.

Kaip buvo paminėta ankščiau, sakome, kad standartinio normaliojo pasiskirstymo tankis yra mažiausiai struktūrizuotas ir siekiame surasti nuokrypius nuo normalumo, pasireiškiančius pagrindinėje skirstinio dalyje, o ne jo uodegose. Tam atliekame projekcijos transformaciją

$$R = 2\Phi(X) - 1 \tag{1.8}$$

kur $\Phi(X)$ yra standartinio normaliojo pasiskirstymo funkcija.

$$\Phi(X) = (1/\sqrt{2\pi}) \int_{-\infty}^{X} e^{-t^2/2} dt \tag{1.9}$$

(1.8) išraiškos pagalba apskaičiuojamas surastos projekcijos skirstinio tankio nuokrypis nuo standartinio Gauso skirstinio tankio. Reikėtų atkreipti dėmesį, kad R įgyja reikšmes iš intervalo $-1 \leq R \leq 1$ ir jei X turi standartinį normalųjį pasiskirstymą, tuomet R bus tolygiai pasiskirstęs šiame intervale. Konkrečiai:

$$p_R(R) = \frac{1}{2} p_\alpha \left[\Phi^{-1}\left(\frac{R+1}{2}\right)\right] / g\left[\Phi^{-1}\left(\frac{R+1}{2}\right)\right] \tag{1.10}$$

čia $g(X)$ - standartinio normaliojo skirstinio tankis. Taigi, R tolygumo intervale matas atitinka X normalumo matą. Tolygumo intervale matu imame funkcijų skirtumo integralą, t.y. atstumą tarp R tikimybinio skirstinio tankio $p_R(R)$ ir tolygiojo skirstinio tankio $p_U(R) = \frac{1}{2}$ intervale $-1 \leq R \leq 1$:

$$\int_{-1}^{1} \left[p_R(R) - \frac{1}{2}\right]^2 dR = \int_{-1}^{1} p_R^2(R)dR - \frac{1}{2} \quad \textbf{(1.11)}$$

Projektavimo indeksą $I(\alpha)$ imame kaip **(1.11)** išraiškos momentų aproksimaciją. Išskleidus $p_R(R)$ Ležandro polinomais gauname:

$$\int_{-1}^{1} p_R^2(R)dR - \frac{1}{2} = \int_{-1}^{1} \left[\sum_{j=0}^{\infty} a_j P_j(R)\right] p_R(R)dR - \frac{1}{2} \quad \textbf{(1.12)}$$

(1.12) išraiškoje nežinomi dydžiai $P_j(R)$ yra Ležandro polinomai. Pirmosios dvi Ležandro polinomų išraiškos:

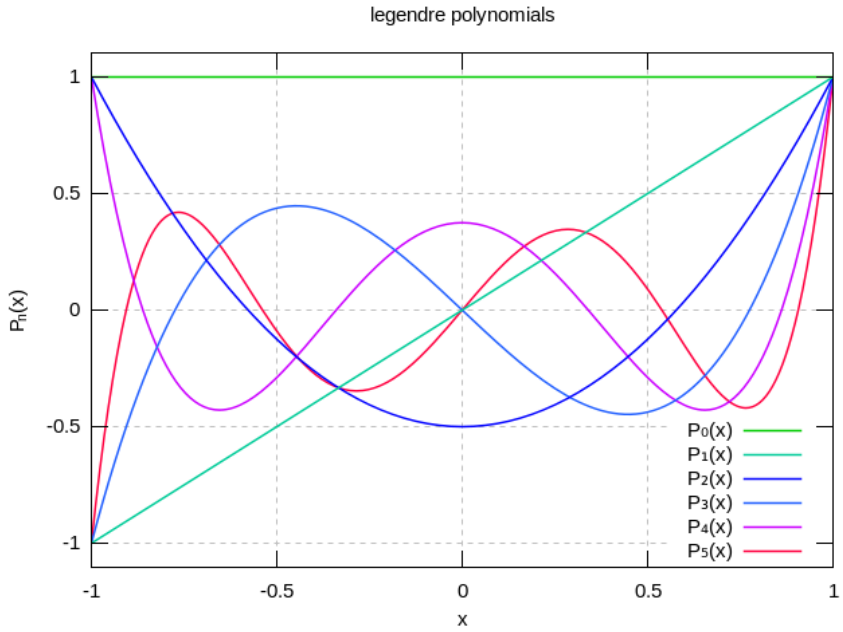
$$\begin{aligned} P_0(R) &= 1 \\ P_1(R) &= R \end{aligned} \quad \textbf{(1.13)}$$

(1.13) išraiškoje nusakytas nulinės ir pirmos eilės Ležandro polinomų suradimas, antros ir aukštesnių eilių, t.y. kai $j \geq 2$ Ležandro polinomų radimui pateikiama bendroji formulė:

$$P_j(R) = \left[(2j-1)RP_{j-1}(R) - (j-1)P_{j-2}(R)\right]/j \quad \textbf{(1.14)}$$

Kad būtų lengviau įsivaizduoti kaip atrodo Ležandro polinomai, kai $j \geq 2$ pateikiamos pirmųjų eilių jų išraiškos, taip pat 1.1 paveiksle matome grafinį Ležandro polinomų vaizdavimą iki $j = 5$ eilės.

$$\begin{gathered} P_n(R) \\ 1 \\ R \\ \frac{1}{2}(3R^2 - 1) \\ \frac{1}{2}(5R^3 - 3R) \\ \frac{1}{8}(35R^4 - 30R^2 + 3) \\ \frac{1}{8}(63R^5 - 70R^3 + 15R) \\ \frac{1}{16}(231R^6 - 315R^4 + 105R^2 - 5) \\ \dots \end{gathered}$$



1.1 pav. Ležandro polinomų grafinis vaizdavimas iki $j = 5$ eilės

Koeficientai a_j gaunami tokiu būdu:

$$a_j = \frac{2j+1}{2}\int_{-1}^{1} P_j(R)p_R(R)\,dR = \frac{2j+1}{2}E_R\left[P_j(R)\right] \qquad \textbf{(1.15)}$$

Todėl

$$\int_{-1}^{1} p_R^2(R)dR - \frac{1}{2} = \sum_{j=1}^{\infty}(2j+1)E_R^2\left[P_j(R)\right]/2 \qquad \textbf{(1.16)}$$

Tolygiajam skirstiniui U intervale $(-1, 1)$, o vidurkis $E\left[P_j(R)\right] = 0$, kai $j > 0$.

Projekcijos indekso išraiška gaunama sumuojant **(1.16)** iki J:

$$I(\alpha) = \sum_{j=1}^{J}(2j+1)E_R^2\left[P_j(R)\right]/2 \qquad \textbf{(1.17)}$$

Pastebėsime, kad **(1.17)** išraiškoje esantis projektavimo indeksas įvertina nuokrypį nuo normalumo, net jei J-osios eilės Ležandro polinomų skleidinys nėra tiksli R tikimybinio skirstinio tankio $p_R(R)$ aproksimacija, nes jis įgauna minimalią reikšmę – 0, kai X yra pasiskirstęs pagal normalųjį skirstinį, o R skirstinys yra tolygusis. Bet kuris X skirstinys, po **(1.8)** transformacijos atlikimo, tampa skirstiniu, kurio tankis yra $p_R(R)$ su nulinėmis reikšmėmis pirmiesiems J Ležandro polinomų momentams, taip pat tai bus mažiausiai įdomus skirstinys.

Praktiškai pritaikant tikslinio projektavimo algoritmą turimai duomenų imčiai, vidurkius **(1.17)** išraiškoje pakeičiame į atitinkamus imties aritmetinius vidurkius, iš **(1.7)** ir **(1.8)** išraiškų, gauname projektavimo indekso įvertį:

$$\hat{I}(\alpha) = \frac{1}{2}\sum_{j=1}^{J}(2j+1)\left[\frac{1}{N}\sum_{i=1}^{N}P_j(2\Phi(\alpha^T z_i) - 1\right]^2 \quad \textbf{(1.18)}$$

Šią išraišką maksimizuojame atsižvelgiant į α vektoriaus q komponentes, su **(1.19)** apribojimu:

$$\alpha^T\alpha = 1 \quad \textbf{(1.19)}$$

Pagal **(1.18)** išraišką projekcijos indeksą apskaičiuojame gana greitai, o visus Ležandro polinomus iki J-osios eilės gauname rekursinės formulės **(1.14)** pagalba.

Efektyvesnei optimizacijai naudinga turėti apskaičiuotas tikslo funkcijos išvestines. Tai galime gana lengvai įvykdyti projektavimo indekso išvestinės skaičiavimui naudojant diferencijuotas Ležandro polinomų išraiškas, kurias gavome rekursinės formulės pagalba:

$$\frac{\partial I}{\partial \alpha_k} = \frac{2}{\sqrt{2\pi}}\sum_{j=1}^{J}(2j+1)E\big[P_j(R)\big]E[P_j'(R)e^{-\frac{X^2}{2}}(Z_k - \alpha_k X)] \quad \textbf{(1.20)}$$

čia X gautas iš **(1.7)** išraiškos ir R iš **(1.8)**. Kiekvieno Ležandro polinomo išvestinę pagal argumentą gana lengvai apskaičiuojame naudodamiesi rekursine formule:

$$\begin{gathered} P_1'(R) = 1 \\ P_j'(R) = PP_{j-1}'(R) + jP_{j-1}(R), \quad j > 1 \end{gathered} \quad \textbf{(1.21)}$$

Projektavimo indekso išvestinės skaičiavimui taikome apribojimą $\alpha^T\alpha = 1$, išlaikant gradiento vektorių $\nabla_\alpha I(\alpha)$ ortogonalų apribojimų funkcijos gradientui $\nabla_\alpha(\alpha^T\alpha) = 2\alpha$. Šiuo tikslu **(1.20)** išraiškoje antrajame vidurkyje iš Z_k atimame $\alpha_k X$. Projekcijos indekso įverčio išvestines $\hat{I}(\alpha)$ **(1.18)** išraiškoje apskaičiuojame vietoje tikėtinų reikšmių imant imčių aritmetinius vidurkius.

Galimas tik vienas parametras susijęs su projekcijos indeksu, kurį nurodo vartotojas - Ležandro polinomų skleidinio eilės J parinkimas. Naudojant Ležandro polinomų eilę – keičiant J reikšmę, galime keisti aproksimavimo tolydumą. Remiantis tikslinio projektavimo algoritmo sudarytojų patirtimi [18], Ležandro polinomų eilę J rekomenduojama rinktis iš intervalo ($4 \le J \le 8$). Taip pat siūloma, kad Ležandro polinomų eilės J reikšmė turi didėti priklausomai nuo turimos imties dydžio. Autorius taip pat pastebi, kad atliekamų skaičiavimų trukmė tiesiškai didėja didinant Ležandro polinomų eilę J vienos dimensijos tikslinio projektavimo algoritmo projekcijos indeksui.

1.2.3. STRUKTŪROS PANAIKINIMAS

Naudojantis **(1.18)** išraiškoje pateiktu projektavimo indekso įverčiu, ieškoma „įdomių“ duomenų projekcijų. Dažniausiai neužtenka vienos projekcijos suradimo, kad būtų pakankamai tiksliai įvertintas daugiamatis tankis; todėl turime surasti daugiau „įdomių‘ krypčių.

Vertinant tankį tikslinio projektavimo metodu yra naudojamas duomenų struktūros panaikinimas, t.y. atliekamas mastelio pakeitimas rasta projektavimo kryptimi taip, kad transformuotų duomenų skirstinys joje tampa normalusis, o tai reiškia, kad yra visiškai „neįdomus“. Tokiu būdu atlikus duomenų imties pakeitimus, ieškant sekančios projektavimo krypties, kurioje duomenų skirstinio tankio kreivė yra labiausiai besiskirianti nuo normaliojo Gauso skirstinio, nebus surasta jau ankščiau aptikta kryptis.

Taigi optimizavimo algoritmo tikslas – rasti maksimalią projekcijos indekso reikšmę, kuri parodo, kad projekcija yra „įdomi“. Manoma, kad surastoji projekcija pateiks informatyvų daugiamačio tankio vaizdą, tačiau dažniausiai nuokrypis nuo normalumo pasireiškia keliose vienamatėse duomenų projekcijose, todėl itin svarbu, kad tikslinio projektavimo algoritmo procedūra rastų kaip įmanoma daugiau informatyvių vienamačių projekcijų.

P. J. Huber pasiūlė platų duomenų struktūros panaikinimo metodikų pasirinkimą [13]. Tačiau iš visų pasiūlytų metodų labiausiai sistemingas J. H. Friedman, W. Stuetzle, A. Schroeder 1984 metais pasiūlytas metodas [17], tinkantis vienamačiam tiksliniam projektavimui.

Procedūros principas - suradus „įdomią“ projekciją, kurios duomenų skirstinio tankis yra labiausiai besiskiriantis nuo normaliojo Gauso skirstinio tankio, šioje kryptyje projektavimo indekso reikšmė yra didžiausia - duomenų struktūra panaikinama, kad nebūtų dar kartą rasta ta pati projekcija, tuomet vėl iš naujo maksimizuojamas projektavimo indeksas. Tokia veiksmų seka iteratyviai kartojama, kol projektavimo indekso reikšmė tampa labai maža, t.y. surastoje kryptyje duomenų skirstinio tankis yra artimas normaliojo Gauso skirstinio tankiui, o surastoji kryptis laikoma „neįdomia“. Pateiksime paprastą procedūrą duomenų struktūrai panaikinti, kuria remiantis skaičiavimai atliekami labai greitai.

Pagal projekcijos indekso apibrėžimą, suprojektuoto tankio vaizdas yra labiausiai „neįdomus“, jei jo pasiskirstymas yra standartinis normalusis. Tokiu atveju duomenų struktūrą panaikiname atlikdami transformacijas, po jų atlikimo tankis įgyja standartinį normalųjį pasiskirstymą. Taip pat reikalaujama transformuoti q kintamuosius, rezultatas pavirsta standartiniu normaliuoju skirstiniu projektuojamame poerdvyje, tačiau palieka visas ortogonalias projektavimo kryptis nepakeistas.

Tarkime, kad $X = \alpha^T Z$ yra vienamatė projekcija su dispersija $var(X) = 1$ ir pasiskirstymo funkcija $F_\alpha(X)$. Tuomet taikoma transformacija:

$$X' = \Phi^{-1}\left(F_\alpha(X)\right) \quad \textbf{(1.22)}$$

Po **(1.22)** išraiškoje nurodytos transformacijos atlikimo, X tampa standartiniu normaliuoju X' pasiskirstymu. Čia Φ^{-1} atvirkščias skirstinys standartinei normaliajai pasiskirstymo funkcijai **(1.9).**

Tegu U yra ortonormuota kvadratinė matrica $(q \times q)$, kur α yra jos pirmoji eilutė. Tuomet taikant tiesinę transformaciją $T = UZ$ gaunamas pasukimas, toks kad naujoji pirmoji koordinatė yra $T_1 = \alpha^T Z = X$.

Tegu $\boldsymbol{\theta}$ yra vektorių transformacija su elementais $\theta_1 \dots \theta_q$, paverčianti T_1 į standartinį normalųjį pasiskirstymą ir $T_2 \dots T_q$ yra vienetinė transformacija:

$$\begin{gathered} \theta_1(T_1) = \Phi^{-1}\left(F_\alpha(T_1)\right) \\ \theta_j\left(T_j\right) = T_j, 2 \le j \le q \end{gathered} \quad \textbf{(1.23)}$$

Tuomet transformacija

$$Z' = U^T \boldsymbol{\theta}(UZ) \quad \textbf{(1.24)}$$

pakeičia projekciją $X = \alpha^T Z$ į standartinį normalųjį pasiskirstymą paliekant visas ortogonalias projektavimo kryptis nepakeistas. Kartojame projekcijos indekso maksimizavimo procedūrą, remdamiesi Z' duomenimis gautais po atlikto struktūros panaikinimo pagal **(1.24)** išraišką.

1.2.4. TANKIO VERTINIMAS

Tikslinio projektavimo algoritmo strategija, aprašyta **1.2.2. Projektavimo indeksas**, **1.2.3. Struktūros panaikinimas** skyriuose, apima tikimybinio tankio $p(Z)$ mažiausiai artimo normaliajam Gauso skirstiniui krypties $p_{\alpha_1}(\alpha_1^T Z)$ suradimą, maksimizuojant nuokrypio nuo normalumo matą – projekcijos indeksą, pateiktą **(1.17)** išraiškoje.

(1.25) išraiškoje gaunamas daugiamačių duomenų tankis po pirmojo struktūros panaikinimo projektavimo kryptimi:

$$p_1(Z) = p(Z) g(\alpha_1^T Z) / p_{\alpha_1}(\alpha_1^T Z) \quad \textbf{(1.25)}$$

čia $Z = \left(\frac{1}{\sqrt{D_j}}\right) U(Y - EY)$ – pradiniai sferinti duomenys, o $p(Z)$ – jų tankis, $g(\alpha_1^T Z)$ – standartinio Gauso skirstinio tankis, $p_{\alpha_1}(\alpha_1^T Z)$ – pirmosios rastos projektavimo krypties tankis.

Ieškoma duomenų tankio išraiškos po antrojo struktūros panaikinimo projektavimo kryptimi:

$$p_2(Z) = p_1(Z) g(\alpha_2^T Z) / p_{\alpha_2}^{(1)}(\alpha_2^T Z) = p(Z) \frac{g(\alpha_1^T Z) g(\alpha_2^T Z)}{p_{\alpha_1}(\alpha_1^T Z) p_{\alpha_2}^{(1)}(\alpha_2^T Z)} \quad \textbf{(1.26)}$$

čia $g(\alpha_2^T Z)$ – standartinio Gauso skirstinio tankis, $p_{\alpha_2}^{(1)}(\alpha_2^T Z)$ yra tolygus marginalusis arba ribinis, antrosios duomenų projektavimo krypties α_2^T, tankis bendrajame tankyje $p_1(Z)$. K-tojoje iteracijoje tankio formulė užrašoma tokiu pavidalu:

$$p_K(Z) = p(Z) \prod_{k=1}^{K} \frac{g(\alpha_k^T Z)}{p_{\alpha_k}^{(k-1)}(\alpha_k^T Z)} \qquad \textbf{(1.27)}$$

Dydis $p_{\alpha_k}^{(k-1)}(\alpha_k^T Z)$ vardiklyje yra marginalusis (ribinis) $\alpha_k^T Z$ tankis, bendrajame pasiskirstyme $p_{k-1}(Z)$ su $p_0(Z) = p(Z)$.

Tam tikru iteracinio proceso metu, tikslinio projektavimo algoritmas nebegali rasti projekcijos, kuri labai skirtųsi nuo normaliojo Gauso skirstinio, tuomet gauname, kad $p_K(Z)$ yra labai panašus į daugiamatį normalųjį Gauso pasiskirstymą. Tuomet kaip tankio aproksimaciją imame:

$$\bar{p}(Z) = g(Z) \prod_{k=1}^{K} \frac{p_{\alpha_k}^{(k-1)}(\alpha_k^T Z)}{g(\alpha_k^T Z)} \qquad \textbf{(1.28)}$$

su

$$g(Z) = \frac{1}{(2\pi)^{q/2}} e^{(-Z^T Z/2)} \qquad \textbf{(1.29)}$$

Suprojektuoti vienamačiai tankiai $p_{\alpha_k}^{(k-1)}$ aproksimuojami naudojant Ležandro polinomų skleidinius:

$$p_{\alpha_k}^{(k-1)}(\alpha_k^T Z) = g(\alpha_k^T Z) \sum_{j=0}^{J} (2j+1) E_{k-1}\left[P_j(R_k)\right] P_j(R_k) \qquad \textbf{(1.30)}$$

kur $R_k = 2\Phi(\alpha_k^T Z) - 1$ ir $E_{k-1}(\cdot)$ yra $p_{k-1}(Z)$ vidurkis. Pakeitus į **(1.28)** išraišką, gaunama daugiamačio tankio aproksimacija:

$$\tilde{p}(Z) = g(Z) \prod_{k=1}^{K} \left[\sum_{j=0}^{J} (2j+1) E_{k-1}\left[P_{jk}\right] P_j(2\Phi(\alpha_k^T Z) - 1) \right] \qquad \textbf{(1.31)}$$

kur $E_{k-1}\left[P_{jk}\right]$ yra $p_{k-1}(Z)$ vidurkis. Tankio įvertis gaunamas įvertinant $E_{k-1}\left[P_{jk}\right]$ pagal transformuotų kintamųjų imties vidurkius.

$$Z_{k-1} = U_{k-1}^T \theta_{k-1}(U_{k-1} Z_{k-2}) \qquad \textbf{(1.32)}$$

Z_{k-1} gautas naudojantis duomenų struktūros panaikinimo procedūra iš **(1.24)** išraiškos. Taigi

$$\hat{E}_{k-1}[P_{jk}] = \frac{1}{N}\sum_{i=1}^{N} P_j\left[2\Phi\left(\alpha_k^T z_{(k-1)i}\right) - 1\right] \quad \textbf{(1.33)}$$

Čia $Z_0 = Z$ - originalūs duomenys.

Šį tankio įvertį stipriai įtakoja pagrindinės skirstinio dalies duomenys, todėl gaunami prasti įvertiniai skirstinio uodegose, tai **(1.8)** transformacijos rezultatas.

Skirstiniai turintys ilgas uodegas, lyginant su standartiniu normaliuoju skirstiniu, sudaro aštrius šuolius transformuoto tankio $p_R(R)$ intervalo pabaigoje, to neužfiksuoja žemos Ležandro polinomų eilės išraiškos $(4 \le J \le 8)$. Tokiu būdu projekcijos indeksas $I(\alpha)$ **(1.17)** ir jo įvertis **(1.18)** yra robastiškas ilgų uodegų atveju. Taip pat, projekcijos indeksas bus linkęs neaptikti sprendinių, kuriems suprojektuotas tankis neturės kitokios struktūros negu ilgos uodegos. Todėl tankio įvertis pateiktas **(1.31)** ir **(1.33)** išraiškose skirs dėmesį tankio svyravimams centrinėje skirstinio dalyje ir aproksimuos skirstinio uodegas normaliojo skirstinio uodegomis. Sukurtojo tikslinio projektavimo algoritmo autorius [16] teigia, kad nėra jokio kitokio metodo kuris pateiktų tikslius daugiamačio tankio įverčius skirstinio uodegose.

1.2.5. OPTIMIZAVIMO STRATEGIJA

Metodika naudojama ieškant maksimalios projekcijos indekso reikšmės stipriai įtakoja ir statistines ir skaičiuojamąsias tikslinio projektavimo algoritmo procedūras. Metodo statistinė galia atsispindi jo gebėjime (duotam imties dydžiui ir duomenų dimensijai) rasti projekcijos indekso maksimalią reikšmę.

Buvo pastebėta, kad lokalių maksimumų atsirandančių dėl imčių nepastovumo, gali būti labai daug. Todėl gali atsitikti taip, kad tikslinio projektavimo algoritmas neras „įdomių“ projekcijų. Šie pseudomaksimumai gali būti atvaizduojami kaip dažnai pasitaikančios bangos nubrėžtos ant tikslo funkcijos (projekcijos indekso) paviršiaus. Bangų amplitudė auga didėjant dimensijai ir mažėjant turimos imties dydžiui.

Glodžioms tikslo funkcijoms labiausiai tinkantys optimizavimo metodai (greičiausio nusileidimo, jungtinių gradientų, kvazi-Niutono) apima pirmosios išvestinės suradimą. Štai todėl ir buvo iškeltas reikalavimas, sudarant projektavimo indeksą, kad būtų galima greitai apskaičiuoti jo išvestines. Minėtieji optimizavimo metodai labai efektyviai, t.y. greitai ir tiksliai, pradedant paieškas pradžios taške įkalnėje, randa pirmąjį tikslo funkcijos maksimumą. Deja, šiuos metodus pritaikius projektavimo indeksui su bangų fenomenu, labai tikėtina, kad bus surandamas pseudomaksimumas, nebent pradžios taškas yra maksimumo pritraukimo srityje.

Į optimizavimo strategija kurią naudojo J. H. Friedman, J. W. Tukey [18] nebuvo įtrauktas išvestinių ieškojimas ir buvo einama dideliais žingsniais ieškant maksimumo. Tai davė šiek tiek pranašumo prieš pseudomaksimumus, tačiau skaičiavimai užėmė daug laiko.

Tikslinio projektavimo algoritmo sudaryme naudojame sudėtinę optimizavimo strategiją. Pradedame su paprastu optimizatoriumi, einame dideliais žingsniais, tokiu būdu prie maksimumo atkeliaujama labai greitai. Tuomet naudojame gradientinį Kvazi-Niutono metodą greitam konvergavimui į sprendinį.

Maksimizavimo procedūrą pradedame vienamačiam projekcijos indeksui $I(\alpha)$, atsižvelgiant į α vektoriaus q komponentes $(\alpha_1 \dots \alpha_q)$. Pirmasis žingsnis - projekcijos indekso $I(\alpha)$ maksimizavimas pagal koordinačių ašis $\alpha = e_i$ $(1 \leq i \leq q)$. Pastebėsime, kad šios ašys yra pradinių duomenų pagrindinių komponenčių kryptys.

1.3. PRADINIŲ DUOMENŲ GENERAVIMAS

Darbe atliktam tikslumo tyrimui naudotos duomenų imtys buvo gautos generuojant Gauso skirstinių mišinius. Mišinio pasiskirstymo tankis $f(x)$ tenkina lygybę:

$$f(x) = \sum_{i=1}^{M} \omega_i g(x, \mu_i, \Sigma_i) \qquad \textbf{(1.34)}$$

kur x – k-matis tolydus duomenų vektorius, ω_i - i-tojo skirstinio svoris mišinyje; $i = 1, \dots, M$. Mišinio svoriai tenkina sąlygas:

$$\omega_i > 0, \sum_{i\check{z}1}^{M} \omega_i = 1 \qquad \textbf{(1.35)}$$

Taip pat, $g(x, \mu_i, \Sigma_i)$ - normaliojo pasiskirstymo tankio funkcija arba tiesiog – tankis su parametrais: vidurkiu μ_i ir kovariacine matrica Σ_i

$$g(x, \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{\frac{k}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{-\frac{(x-\mu_i)' \Sigma_i^{-1} (x-\mu_i)}{2}} \qquad \textbf{(1.36)}$$

Pilnas Gauso mišinių modelis yra nusakomas vidurkių vektoriumi, kovariacinėmis matricomis ir mišinių svoriais įeinančiais į sudarytąjį modelį. Šiuos parametrus pažymime taip [8]:

$$\lambda = \left\{ \omega_i, \mu_i, \sum\nolimits_i \right\}, i = 1, \dots, M. \qquad \textbf{(1.37)}$$

Darbe tikslumo tyrimui atlikti, buvo generuojami Gauso skirstinių mišiniai su keturiomis klasėmis. Taip pat buvo sudaryti trys skirtingi klasių persidengimo atvejai, naudojami tyrime

kiekvienai turimų duomenų dimensijai: pirmuoju atveju klasės yra nepersidengusios, antruoju – jos yra vidutiniškai persidengusios, trečiuoju atveju – sugeneruotos klasės yra gausiai persidengusios viena su kita.

1.4. PAKLAIDŲ MATAVIMO METRIKOS

Eksperimentiškai nustatyta matuojamo dydžio vertė, skiriasi nuo tikrosios dydžio vertės, tokiu būdu atsiranda paklaidos.

Šiame darbe tikslumo tyrimui naudojant tikslinio projektavimo algoritmą, neparametriškai vertinant daugiamatį tankį buvo pasirinktos tokios paklaidų matavimo metrikos:

$$\int_{R^q} \left(\hat{f}(x) - f(x)\right)^2 f(x)\, dx \qquad \textbf{(1.38)}$$

$$\int_{R^q} (\hat{f}(x) - f(x))^2\, dx \qquad \textbf{(1.39)}$$

čia $f(x)$ - tikroji tankio reikšmė, $\hat{f}(x)$ - tankio įverčio reikšmė.

Pirmajai paklaidų matavimo metrikai empirinis analogas užrašomas taip:

$$\sigma_1 = \frac{1}{N}\sum_{i=1}^{N} \left(f(x_i) - \hat{f}(x_i)\right)^2 \qquad \textbf{(1.40)}$$

antrajai paklaidų matavimo metrikai:

$$\sigma_2 = \frac{1}{N}\sum_{i=1}^{N} \frac{\left(f(x_i) - \hat{f}(x_i)\right)^2}{f(x_i)} \qquad \textbf{(1.41)}$$

Antroji - vidutinė integruota kvadratinė paklaida, jautri paklaidoms skirstinio uodegose. Vertinant centinę skirstinio dalį, o ne uodegas geriau naudoti pirmąją paklaidų matavimo metriką, tačiau palyginimui buvo nagrinėjama ir antroji.

1.5. TESTAI DUOMENŲ NORMALUMUI TIKRINTI

Klasikiniai metodai daugiamatėje analizėje remiasi prielaida, kad duomenys yra gaunami iš daugiamatės normalios populiacijos, tai reiškia, kad jie yra pasiskirstę pagal daugiamatį normalųjį skirstinį. Dėl šios priežasties jie yra itin svarbūs, šie testai sulaukė labai daug dėmesio, todėl juos nagrinėja daugelis autorių. [4]

Taigi norint sužinoti ar turimi duomenys yra pasiskirstę pagal normalųjį skirstinį, tokiai procedūrai atlikti yra naudojami įvairūs normalumo testai. Mūsų atvejis - sudėtingesnis, kadangi duomenys yra daugiamačiai, todėl ir normalumo vertinimo testus turime pasirinkti atitinkamai daugiamačiams duomenims.

Vertinant daugiamačių duomenų normalumo prielaidą, plačiausiai yra naudojami trys normalumo testai, tai Mardia normalumo testas, vertinantis daugiamačius asimetrijos ir eksceso koeficientus, bei Henze - Zirkler ir Royston H normalumo testai daugiamačiams duomenims. [3]

Mardia daugiamačio normalumo testas eksceso koeficiento radimui, skaičiavimo laiko užima proporcingai esamam stebėjimų skaičiui. Priešingai, skaičiavimo laikas Henze - Zirkler normalumo testo, lyginant su Mardia testu ieškančio asimetrijos koeficiento reikšmės, yra apytiksliai proporcingas stebėjimų skaičiaus kvadratui. [2]

Prieš pardedant kiekvieno normalumo testo aprašymą atskirai, padarysime bendras prielaidas, todėl sakome, kad turime N nepriklausomų k-mačių stebėjimų, x_i, $i = 1, \dots, N$. X reiškia, kad $N \times k$ stebėjimų matrica. Norima patikrinti ar šie daugiamačiai stebėjimai yra pasiskirstę pagal normalųjį Gauso skirstinį, $MVN_k(\mu, \Sigma)$. Imties vidurkis yra $\bar{x} = \frac{1}{N}\sum_i x_i$ ir turimos imties kovariacinė matrica $S = \frac{1}{N}\Sigma(x_i - \bar{x})(x_i - \bar{x})'$.

Sekančiuose skyriuose bus aptariamas kiekvienas iš trijų normalumo testų.

1.5.1. MARDIA NORMALUMO TESTAS

Mardia testas remiasi asimetrijos ir eksceso koeficientų skaičiavimu daugiamačiams duomenims. [1] Turint itin didelę stebėjimų imtį daugiamatis Mardia testo asimetrijos koeficientas yra asimptotiškai pasiskirstęs, pagal Chi-kvadrato skirstinį; o daugiamatis Mardia eksceso koeficientas yra pasiskirstęs pagal standartinį normalųjį pasiskirstymą su $\mu = 0$ ir $\Sigma = I$. [7]

Mardia normalumo testo asimetrija ir ekscesas vertinami abu kartu. Hipotezė apie imties normalumą testui yra priimama, jeigu ji priimama tiek pagal asimetrijos, tiek pagal eksceso kriterijus. Trumpiau vadinsime Mardia normalumo testu.

Mardia apibrėžė daugiamačius asimetrijos $b_{1,k}$ ir eksceso $b_{2,k}$ koeficientus taip:

$$b_{1,k} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} g_{ij}^3 \qquad \textbf{(1.42)}$$

$$b_{2,k} = \frac{1}{N}\sum_{i=1}^{n} g_{ii}^2 \qquad \textbf{(1.43)}$$

kur $g_{ij} = (x_i - \bar{x})'S^{-1}(x_j - \bar{x})$. Testo statistika:

$$z_1 = \frac{(k+1)(N+1)(N+3)}{6\{(N+1)(k+1)-6\}} b_{1,k} \tag{1.44}$$

yra aproksimuojama χ^2 pasiskirstymu su $k(k+1)(k+2)/6$ laisvės laipsnių. Testo statistika

$$z_2 = \frac{b_{2,k} - k(k+2)}{\sqrt{8k(k+2)/N}} \tag{1.45}$$

yra aproksimuojama standartiniu normaliuoju $N(0,1)$ pasiskirstymu.

1.5.2. HENZE – ZIRKLER NORMALUMO TESTAS

Henze – Zinkler normalumo tikrinimo testas paremtas empirine charakteristine funkcija. Darant prielaidą, kad kovariacinė matrica S yra nesinguliari, turime:

$$\begin{aligned} T = {} & \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} exp\left\{-\frac{\beta^2}{2}(x_i - x_j)' S^{-1}(x_i - x_j)\right\} - \\ & -2(1+\beta^2)^{-\frac{k}{2}} \sum_{i=1}^{N} exp\left\{-\frac{\beta^2}{2(1+\beta^2)}(x_i - \bar{x})' S^{-1}(x_i - \bar{x})\right\} \\ & + N(1+2\beta^2)^{-\frac{k}{2}} \end{aligned} \tag{1.46}$$

kur

$$\beta = \frac{1}{\sqrt{2}}\left\{\frac{N(2k+1)}{4}\right\}^{\frac{1}{k+4}} \tag{1.47}$$

Kai $N \to \infty$, pirmieji du T momentai – vidurkis ir dispersija užrašomi taip:

$$E(T) = 1 - (1+2\beta^2)^{-\frac{k}{2}}\left\{1 + \frac{k\beta^2}{1+2\beta^2} + \frac{k(k+2)\beta^4}{2(1+2\beta^2)^2}\right\} \tag{1.48}$$

$$\begin{aligned} Var(T) = {} & 2(1+4\beta^2)^{-\frac{k}{2}} \\ & + 2(1+2\beta^2)^{-k}\left\{1 + \frac{2k\beta^4}{(1+2\beta^2)^2} + \frac{3k(k+2)\beta^8}{4(1+2\beta^2)^4}\right\} \\ & - 4w^{-\frac{k}{2}}\left\{1 + \frac{3k\beta^4}{2w} + \frac{k(k+2)\beta^8}{2w^2}\right\} \end{aligned} \tag{1.49}$$

kur $w = (1+\beta^2)(1+3\beta^2)$.

Henze – Zinkler siūlo gauti p-reikšmę iš prielaidos, paremtos atlikus eilę simuliavimų, kad T yra aproksimuojamas lognormaliuoju pasiskirstymu. Todėl $VZ = \ln\{1 + Var(T)/E(T)^2\}$ ir $EZ = \ln\{E(T)\} - \frac{VZ}{2}$. Transformacija $Z = \{\ln(T) - EZ\}/\sqrt{VZ}$. Dydžio Z p-reikšmė apskaičiuojama pagal $p = 2\Phi(-|Z|)$ formulę, kur $\Phi()$ yra pasiskirstymo funkcija. [2]

1.5.3. ROYSTON H TESTAS

Šis testas yra populiaraus vienamačio Shapiro – Wilk W testo modifikacija daugiamačių duomenų atveju. Tarkime, kad W_j yra Shapiro – Wilko statistikos reikšmė k-mačio pasiskirstymo j-tajam elementui, $1 \leq j \leq p$. [5] Tuomet, Royston H testo statistika apibrėžiama taip:

$$R_j = \left\{\Phi^{-1}\left[\frac{1}{2}\Phi\left\{\left(\left(1-\mathrm{W_j}\right)^{\lambda}-\mu\right)/\sigma\right\}\right]\right\}^2 \qquad \textbf{(1.50)}$$

kur λ, μ, σ gaunami turimam N iš 2-osios lentelės Royston [9]; ir $\Phi()$ yra pasiskirstymo funkcija. Jei turimi duomenys yra daugiamačiai, tai:

$$H = \xi \sum \frac{R_j}{p} \qquad \textbf{(1.51)}$$

yra aproksimuojamas $\chi^2_{\hat{\xi}}$ skirstiniu, kur

$$\hat{\xi} = p/[1 + (p-1)\bar{c}] \qquad \textbf{(1.52)}$$

kur $\bar{c}$ – koreliacijos tarp R_j vidurkio įvertis. [6]

1.6. TYRIMO METU NAUDOJAMA PROGRAMINĖ ĮRANGA

Esant didelei baigiamojo darbo tiriamosios dalies apimčiai buvo iškeltas reikalavimas pasirenkamai programinei įrangai, kad sudėtingi skaičiavimai būtų atliekami su mažiausiomis laiko sąnaudomis, o duomenų grafinio vaizdavimo pasirinkimo galimybės būtų plačios. Tam puikiai tiko R statistinis paketas (pavadinimas kilęs iš autorių pirmųjų vardų raidžių – Ross Ihaka ir Robert Gentleman). Tai nemokama, atviro kodo programinė įranga skirta duomenų analizei, statistiniams skaičiavimams atlikti, bei gautiesiems rezultatams grafiškai vaizduoti. R statistinis paketas leidžia patogiai atlikti veiksmus su vektoriais ir matricomis.

Siekiant supaprastinti darbą šiuo statistiniu paketu, buvo instaliuota R vartotojo sąsaja – RStudio. Šią programinę įrangą galima atsisiųsti iš tinklalapių: http://cran.at.r-project.org/; https://www.rstudio.com/ide/download/.

R statistiniame pakete gausus įvairių bibliotekų ir paketų pasirinkimas, taip pat plačios statistinių ir grafinių metodų panaudojimo galimybės; labai gerai ištobulintos pagalbos (angl. Help) priemonės. Taip pat pastebėta, kad daugumos naujausių algoritmų programinės realizacijos dažniausiai pateikiamos naudojant R statistinį paketą.

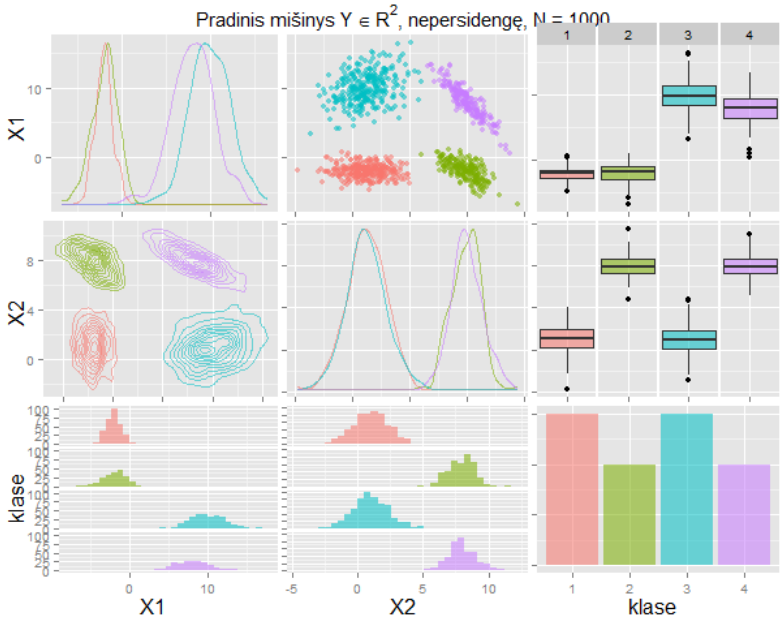
2. TIRIAMOJI DALIS

2.1. MIŠINIŲ SKIRSTINIŲ PERSIDENGIMO LYGMENYS

Tikslumo tyrimui atlikti buvo generuojamos imtys turinčios tris skirtingus persidengimo lygmenis:

1. Nepersidengusios klasės;
2. Vidutiniškai persidengusios klasės;
3. Gausiai persidengusios klasės.

Žemiau esančiuose 2.1 pav., 2.2 pav., 2.3 pav. grafikuose pateikiame visus klasių persidengimo lygmenų pavyzdžius R^2 atveju. Aukštesnių dimensijų - R^3, R^4, R^5 persidengimo lygmenų grafinės iliustracijos pateikiamos 1 priede.

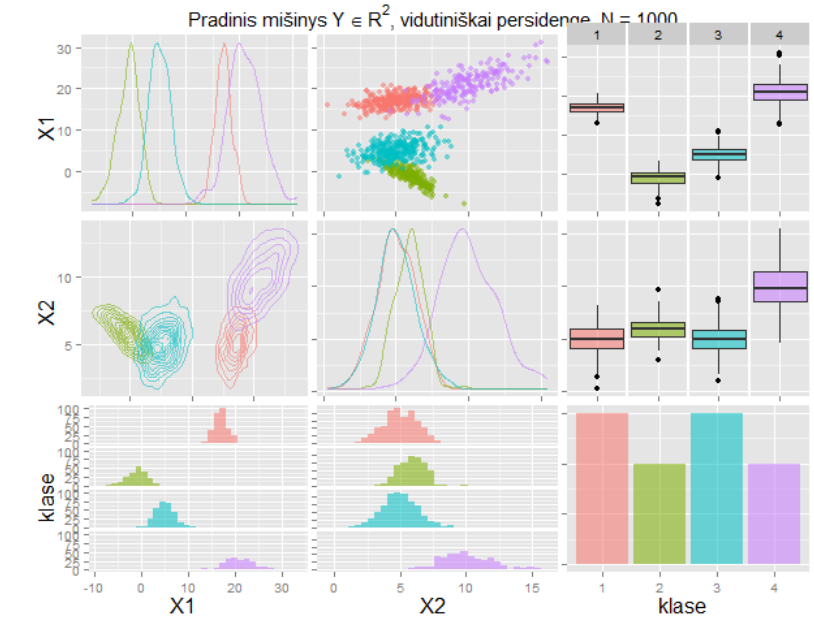


2.1 pav. Nepersidengusių klasių atvejis R^2

2.1 paveikslas suskaidytas į devynias dalis, pirmuose dviejuose pagrindinės įstrižainės dalyse vaizduojami imčių tankiai pagal pirmąją ir antrąją koordinates. Trečiajame stulpelyje vaizduojamos klasių stačiakampės diagramos. Svarbiausia dalis grafike - pirmosios eilutės antrasis paveikslėlis, jame vaizduojamas pradinių imčių išsidėstymas, labai aiškiai matomos visiškai atsiskyrusios keturios klasės. Šias klases sudarančių imčių vidurkiai:

$\mu_1 = (-2; 1)$, $\mu_2 = (-2; 8)$, $\mu_3 = (10; 1)$ ir $\mu_4 = (8; 8)$ ir jų kovariacinės matricos:

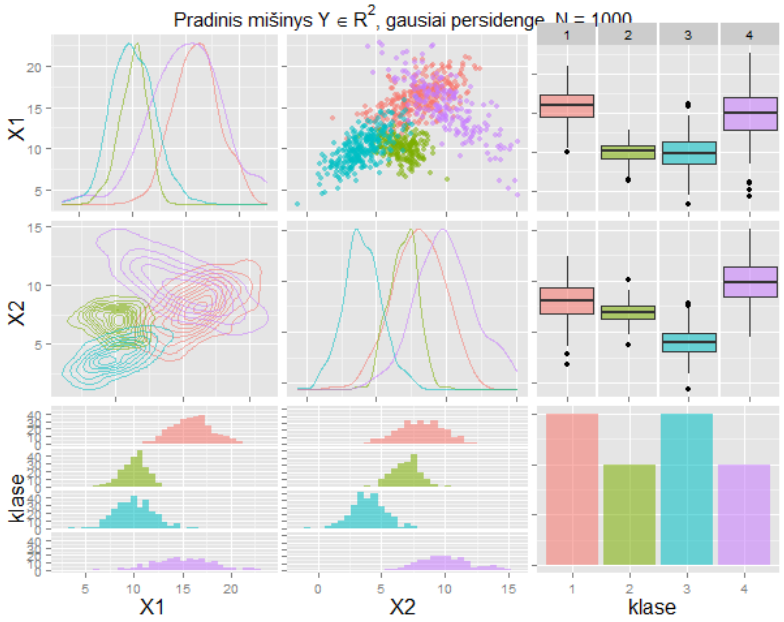
$\sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, $\sigma_2 = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}$, $\sigma_3 = \begin{bmatrix} 5 & 1 \\ 1 & 2 \end{bmatrix}$ ir $\sigma_4 = \begin{bmatrix} 5 & -2 \\ -2 & 1 \end{bmatrix}$.



2.2 pav. Vidutiniškai persidengusių klasių atvejis R^2

2.2 paveiksle vaizduojamos klasės su vidutinišku persidengimo lygmeniu. Šiuo atveju skirstinių vidurkiai: $\mu_1 = (17; 5)$, $\mu_2 = (-1; 6)$, $\mu_3 = (5; 5)$ ir $\mu_4 = (21; 10)$, o kovariacinės matricos:

$\sigma_1 = \begin{bmatrix} 2.5 & 1 \\ 1 & 2.1 \end{bmatrix}$, $\sigma_2 = \begin{bmatrix} 4 & -1.7 \\ -1.7 & 1 \end{bmatrix}$, $\sigma_3 = \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}$ ir $\sigma_4 = \begin{bmatrix} 10 & 4.8 \\ 4.8 & 4 \end{bmatrix}$.

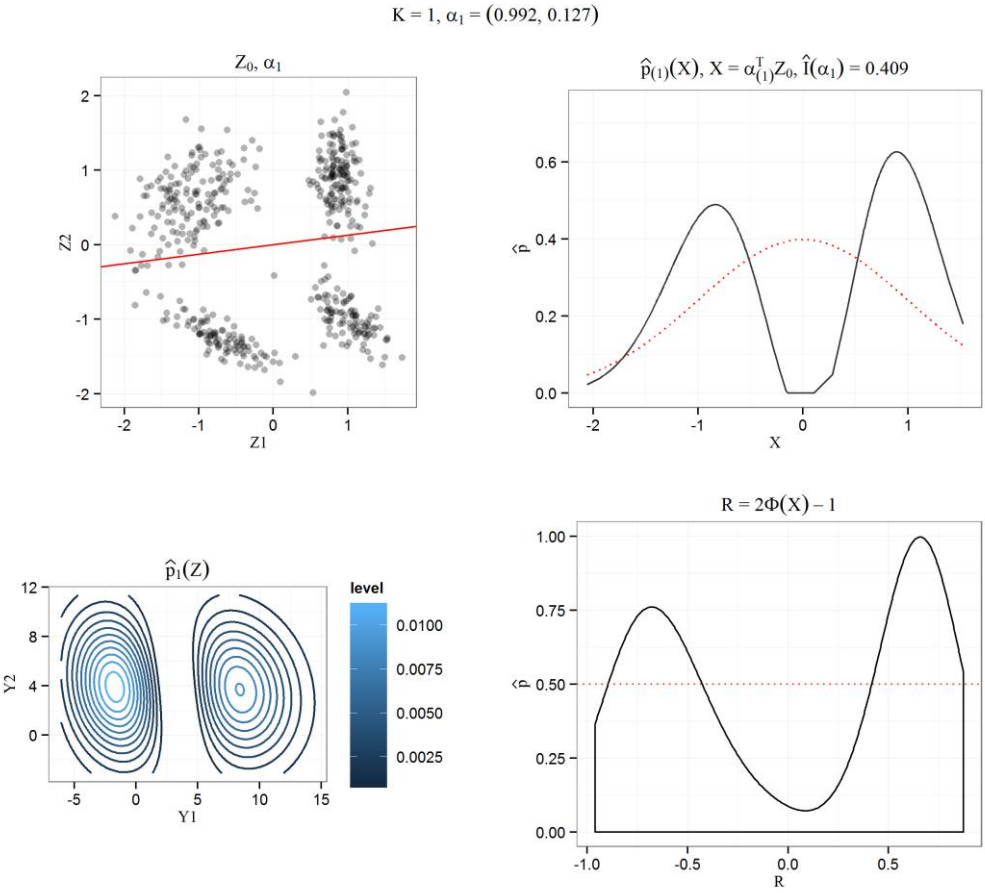


2.3 pav. Gausiai persidengusių klasių atvejis R^2

2.3 pav. vaizduojamos klasės su gausiu persidengimo lygmeniu, jos visos - susiliejusios tarpusavyje. Šių klasių vidurkiai: $\mu_1 = (16; 8)$, $\mu_2 = (10; 7)$, $\mu_3 = (10; 4)$ ir $\mu_4 = (15; 10)$; kovariacinės matricos: $\sigma_1 = \begin{bmatrix} 5 & 3 \\ 3 & 4 \end{bmatrix}$, $\sigma_2 = \begin{bmatrix} 1.5 & -0.4 \\ -0.4 & 1 \end{bmatrix}$, $\sigma_3 = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}$ ir $\sigma_4 = \begin{bmatrix} 10 & -4.8 \\ -4.8 & 4 \end{bmatrix}$.

2.2. TIKSLINIO PROJEKTAVIMO ALGORITMO ILIUSTRACIJA R^2 ATVEJU

Tikslinio projektavimo algoritmo pritaikymą praktiniu atveju iliustruosime R^2 pavyzdžiu - nagrinėjamas mišinyje visiškai atsiskyrusių klasių atvejis.



2.4 pav. Sferinta imtis, rasta pirmoji duomenų projektavimo kryptis

2.4 paveiksle pateikiama tikslinio projektavimo algoritmo pirmoji iteracija. Viršutinėje kairėje dalyje matome sferintų duomenų vaizdą, po atliktos duomenų transformacijos – pasukimo, vietos ir mastelio pakeitimų - turimų duomenų vidurkis $\mu = 0$, o kovariacinė matrica $\Sigma = I$- vienetinė. Turint tokius duomenis, ieškoma „įdomi“ kryptis, kurioje gaunamas labiausiai nuo standartinio Gauso skirstinio tankio besiskiriantis duomenų tankis. Šiuo atveju rasta projektavimo kryptis $\alpha_1 = (0.992; 0.127)$, ją atitinka raudona tiesė.

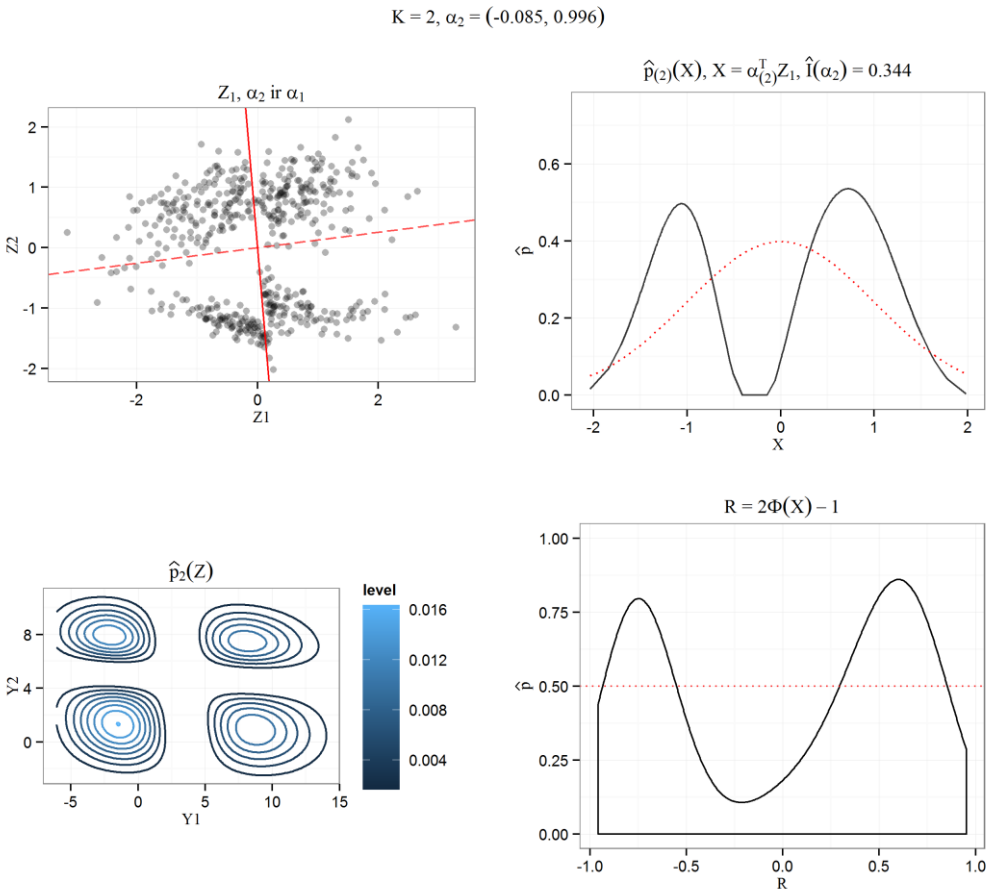
Remiantis skyreliu **1.2.2 Projektavimo indeksas** žinome, kad kuo didesnė projektavimo indekso $\hat{I}(\alpha_1)$ rekšmė, tuo rastoji kryptis yra „įdomesnė“. Viršutiniame kairiajame grafike raudonai pažymėta tiesė ir yra ta projekcija, kurios kryptimi projektuojami visi imties taškai. Po duomenų projektavimo atlikimo šia kryptimi, joje panaikinama duomenų struktūra, t.y. suprojektuoti duomenys jau bus pasiskirstę pagal standartinį Gauso skirstinį ir ieškant optimalios krypties su didžiausia projektavimo

indekso reikšme antrojoje algoritmo iteracijoje, ši kryptis bus vertinama kaip labiausiai „neįdomi“. Tokiu būdu išvengiama pasikartojančio jau ankščiau surastos krypties aptikimo.

2.4 paveikslo viršutinėje dešinėje pusėje juoda ištisinė linija – projekcijos tankis įvertintas Ležandro polinomais, raudona punktyrinė linija – standartinio Gauso skirstinio tankis. Iš šios grafiko dalies, matome, kad suradus pirmąją projektavimo kryptį, duomenų skirstinio tankis - dvi aiškiai atsiskiriančios kalvos. Toks vaizdas labai skiriasi nuo standartinio Gauso skirstinio tankio, tai reiškia, kad surastoji kryptis yra labiausiai „įdomi“. Šiuo atveju projektavimo indekso reikšmė $\hat{I}(\alpha_1) = 0.409$.

Apatinėje kairėje grafiko dalyje matome kontūrus, tai tankis iš vienos, t.y. ką tik surastos krypties.

Apačioje dešinėje – projekcijos transformacija $R = 2\Phi(X) - 1$, parodanti surastos projekcijos skirstinio tankio nuokrypius nuo tolygiojo skirstinio tankio intervale $[-1,1]$. Atlikus pirmąją tikslinio projektavimo algoritmo iteraciją iš šios grafiko dalies, matome, kad nuokrypiai yra itin dideli.

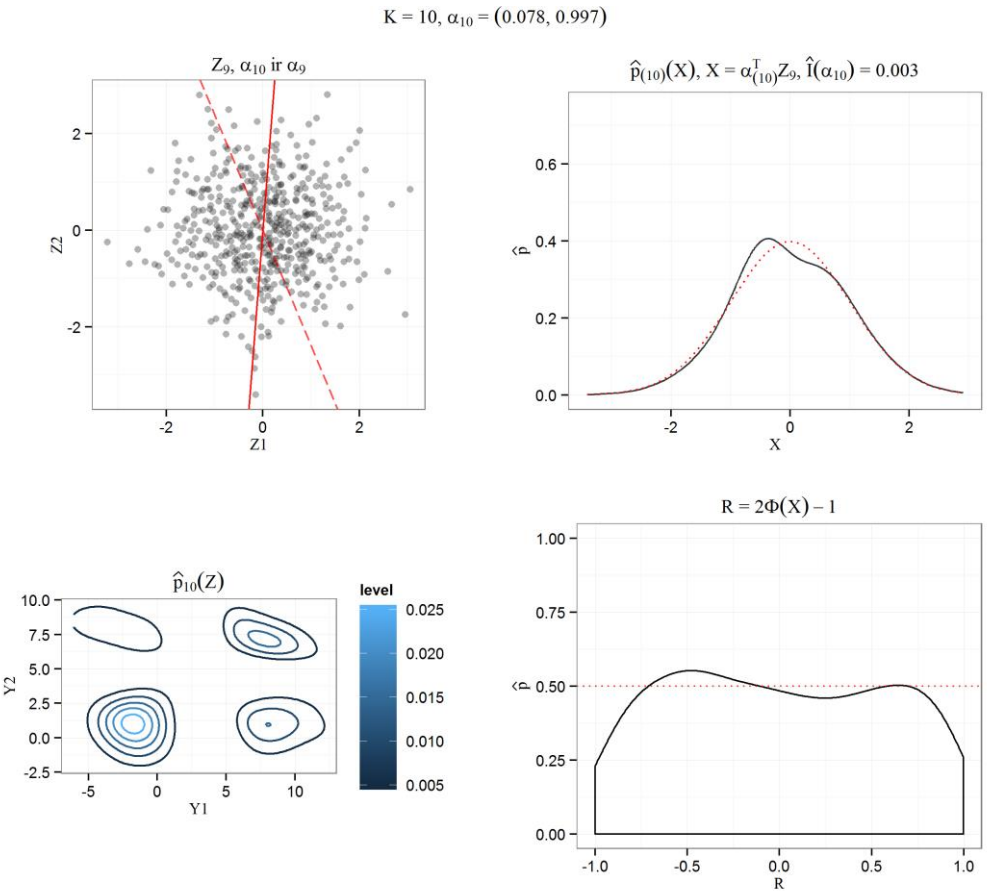


2.5 pav. Sferinta imtis, rasta antroji duomenų projektavimo kryptis

2.5 paveiksle pateikiama antroji tikslinio projektavimo algoritmo iteracija surastai krypčiai $\alpha_2 = (-0.085; 0.996)$. Viršutiniame kairiajame grafike raudona punktyrinė linija - pirmojoje algoritmo iteracijoje surasta duomenų projektavimo kryptis, raudona ištisinė linija – surasta antrojoje iteracijoje, labiausiai nuo standartinio Gauso skirstinio tankio kreivės besiskirianti projekcijos kryptimi

suprojektuotų duomenų skirstinio tankio kreivė. Projektavimo indekso reikšmė šioje kryptyje - $\hat{I}(\alpha_2) = 0.344$.

Šio paveikslo kairėje apačioje esančioje dalyje matome, kad tankio vertinimui naudojant dvi projektavimo kryptis, visiškai atsiskiria keturios klasės. Dešinėje apačioje matome, kad ir po antrosios tikslinio projektavimo algoritmo iteracijos nuokrypiai išlieka akivaizdūs.



2.6 pav. Sferinta imtis, rasta dešimtoji duomenų projektavimo kryptis

Remiantis paklaidų grafikais, R^2 nepersidengusių klasių atveju, daugiau nei dviejų projektavimo krypčių suradimas geresnio rezultato nepateikia.

Išnagrinėjus praleistų sekančių septynių algoritmo iteracijų rezultatus pateikiamus 2 priede, galima teigti, kad projektavimo indekso reikšmė lyginant su pirmąja ir antrąja algoritmo iteracijomis visą laiką mažėja; surastąja kryptimi suprojektuotų duomenų skirstinio tankio forma labai panaši į normaliojo Gauso skirstinio tankį. Tai reiškia, kad surastose sekančiose kryptyse, tarp jų ir 2.6 pav. pavaizduotai projektavimo krypčiai α_{10}, duomenys yra pasiskirstę pagal standartinį Gauso skirstinį. Taigi vienareikšmiškai galima teigti, kad tokios projektavimo kryptys yra visiškai „neįdomios“.

2.3. TANKIO ĮVERTINIO, PAGRĮSTO TIKSLINIU PROJEKTAVIMU, TYRIMAS

Magistro baigiamojo darbo daugiamačio tankio įvertinio, pagrįsto tiksliniu projektavimu, tyrimui buvo sugeneruota po 100 imčių su keturiomis skirtingomis iš eilės einančiomis dimensijomis: R^2, R^3, R^4 ir R^5; tūriais N: 100, 200, 500, 1000 imties taškų. Kiekvienai imčiai buvo atliekamas tikslinis projektavimas, taip sukonstruojamas tankio įvertinys. Kiekvienam tankio įvertiniui buvo sugeneruojama po naują imtį naudojant tuos pačius mišinio parametrus, kuriais generuota jo tiriamoji imtis, tačiau pasirenkant tūrį $N = 5000$. Pasirinkdami didelį imties tūrį siekiame išvengti atsitiktinumų ir gauti stabilesnes paklaidas atliekant tankio vertinimą.

Daugiamačiam tankiui įvertinti naudojamos skyriuje **1.4. Paklaidų matavimo metrikos** nurodytos metrikos. Žemiau pateikiamuose grafikuose įvedamas normuotos paklaidos žymėjimas:

$$\sigma_i'(K) = \frac{\sigma_i(K)}{\sigma_i(0)}, i = 1,2$$

t.y. paklaidos reikšmė K-tojoje projektavimo kryptyje normuojama dalinant iš paklaidos reikšmės gautos aproksimuojant tankį standartiniu Gauso skirstiniu.

Turint absoliučias paklaidų reikšmes susiduriama su problema, negalime jų palyginti esant skirtingai dimensijai ar imties tūriui. Šiai problemai spręsti yra skaičiuojamos santykinės paklaidos, t.y. tankio vertinimo paklaidos visose projektavimo kryptyse yra lyginamos su teoriškai optimalaus krypčių kiekio paklaida, t.y. kur gaunama teoriškai minimali paklaida. Taigi, trumpai tariant santykinė paklaida - šių dviejų paklaidų santykis, taip procentiškai įvertinamas jos „blogumas“, lyginant su optimalaus krypčių kiekio paklaida.

2.3.1. TANKIO ĮVERTINIO TIKSLUMO TYRIMAS, ALGORITMO STABDYMUI, NAUDOJANT PROJEKTAVIMO INDEKSĄ

Tyrimui aprašyti naudojami skyriuje **2.3. Tankio įvertinio, pagrįsto tiksliniu projektavimu, tyrimas** nurodyti parametrai. Siekiame parinkti optimalų projektavimo krypčių kiekį, kad galėtume įvertinti kada palankiausia būtų stabdyti algoritmą, nes sekančios krypties suradimas gaunamo rezultato nepagerintų ir duomenys bet kokiu atveju jau būtų pasiskirstę pagal standartinį normalųjį skirstinį. Tam tikslui įvedame naują dydį – slenkstį - tam tikrą projektavimo indekso lygmenį, kuriuo vertiname ar projektavimo indekso reikšmė K-tojoje kryptyje viršija šio lygmens reikšmę, o $K + 1$-ojoje kryptyje yra žemiau lygmens reikšmės. Naudodamiesi slenksčio pasirinkimo galimybe, galime įvertinti, kad nuo surastos K- tosios krypties sekančiose projektavimo kryptyse duomenys bus

pasiskirstę pagal standartinį Gauso skirstinį. Projektavimo krypčių kiekį, kuris tenkina projektavimo indekso slenksčio sąlygą žymime K^*.

Taip pat vertiname daugiamačio tankio paklaidas visose projektavimo kryptyse, gautas pagal pirmąją ir antrąją paklaidų matavimo metrikas **(1.40)** ir **(1.41)** išraiškose. Kiekvienam parametrų (dimensijos, persidengimo lygmens, tūrio, Ležandro polinomų eilės) deriniui imame santykį $\sigma_i^*, (i = 1, 2)$: tankio vertinimo, kuriam naudojama nuo 0 iki 25 rastų projektavimo krypčių, 100 normuotų i-tųjų paklaidų vidurkio reikšmes daliname iš mažiausio 100 normuotų i-ųjų paklaidų vidurkio. Šį santykį užrašome taip:

$$\sigma_i^*(K) = \frac{\frac{1}{100}\sum_{j=1}^{100}\sigma_i^{(j)}(K)}{\min\limits_K \frac{1}{100}\sum_{j=1}^{100}\sigma_i^{(j)}(K)}, \quad i = 1,2$$

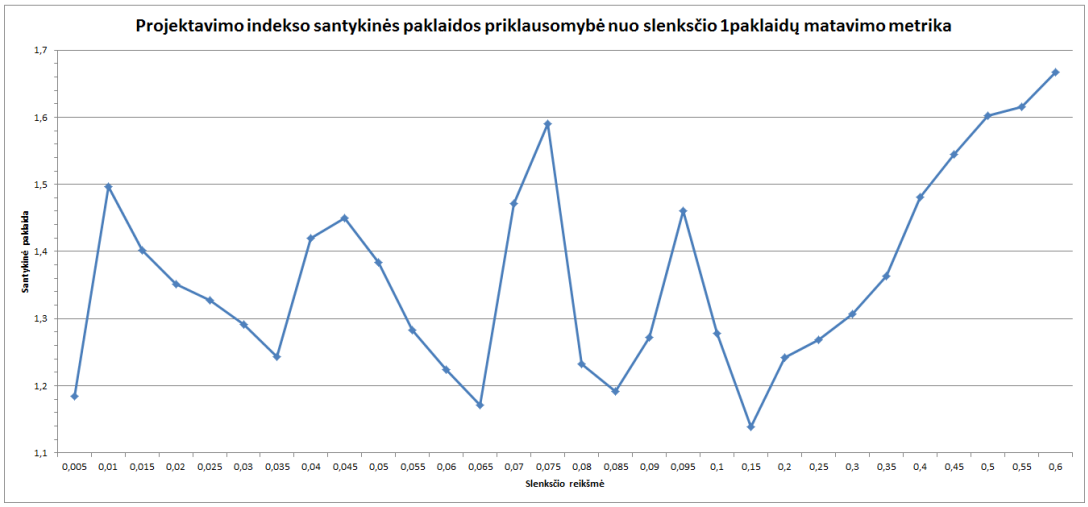
Projektavimo krypčių kiekį, su kuriuo santykis $\sigma_i^*(K)$ lygus vienetui vadiname optimaliu. Galima išskirti du atvejus, kai projektavimo krypčių kiekis parenkamas naudojant projektavimo indekso slenkstį:

1. Tankio vertinimui naudojama mažiau krypčių negu optimalus krypčių kiekis;
2. Tankio vertinimui naudojama daugiau krypčių negu optimalus krypčių kiekis.

Krypčių kiekį, kuris buvo parinktas naudojant projektavimo indekso slenkstį žymime K^*. Kiekvienam imties tūriui imame likusių parametrų (dimensijos, persidengimo lygmens, Ležandro polinomų) $\sigma_i^*(K^*)$, $i = (1, 2)$ vidurkį.

Paveiksluose vaizduojama ką tik minėtų daugiamačio tankio vertinimo paklaidų santykių $\sigma_i^*(K^*)$ vidurkių priklausomybė nuo pasirenkamo projektavimo indekso slenksčio reikšmės.

Slenksčio reikšmių pasirinkimo intervalas $[0{,}005;\ 0{,}6]$.



2.7 pav. Santykinės paklaidos $\sigma_1^*(K^*)$ priklausomybė nuo slenksčio, $N = 100$

2.8 pav. Santykinės paklaidos $\sigma_2^*(K^*)$ priklausomybė nuo slenksčio, $N = 100$

Daugiamatį tankį įvertinus pagal abi paklaidų matavimo metrikas, gautuose grafikuose pastebime, kad esant mažam imties tūriui, šiuo atveju $N = 100$, paklaidų santykiai yra jautrūs pasirenkamo slenksčio reikšmei. Trijose vietose gaunamos itin išsiskiriančios reikšmės. Mažiausia santykinė paklaida abiem atvejais gaunama su slenksčio reikšme lygia 0,15. Taigi esant mažam imties tūriui atsiranda didelis santykinių paklaidų $\sigma_i^*(K^*)$, $i = 1, 2$ svyravimas.

2.9 pav. Santykinės paklaidos $\sigma_1^*(K^*)$ priklausomybė nuo slenksčio, $N = 200$

2.9 pav. mažiausia santykinė paklaida pirmąja paklaidų matavimo metrika, gaunama su slenksčio reikšme lygia 0,045, yra 8,37%.

2.10 pav. Santykinės paklaidos $\sigma_2^*(K^*)$ priklausomybė nuo slenkščio, $N = 200$

2.10 pav. mažiausia santykinė paklaida antrąja paklaidų matavimo metrika, gaunama su slenkščio reikšme lygia 0,1 yra 9,08%.

2.11 pav. Santykinės paklaidos $\sigma_1^*(K^*)$ priklausomybė nuo slenkščio, $N = 500$

2.11 pav. mažiausia santykinė paklaida, naudojant daugiamačio tankio paklaidų vertinimui pirmąja paklaidų matavimo metriką, gaunama su slenkščio reikšme lygia 0,035, yra 6,41%.

2.12 pav. Santykinės paklaidos $\sigma_2^*(K^*)$ priklausomybė nuo slenksčio, $N = 500$

2.12 pav. mažiausia santykinė paklaida su antrąja paklaidų matavimo metrika, esant slenksčio reikšmei lygiai 0,04, yra 5,82%.

Apibendrinant 2.11 pav. ir 2.12 pav., turint pakankami didelį imties tūrį - $N = 500$, mažiausia santykinė paklaida gaunama parinkus panašias slenksčio reikšmes: pirmuoju atveju pakanka – 0,035, antruoju – 0,04.

2.13 pav. Santykinės paklaidos $\sigma_1^*(K^*)$ priklausomybė nuo slenksčio, $N = 1000$

2.13 pav. mažiausia santykinė paklaida, naudojant pirmąja paklaidų matavimo metriką, gaunama su slenksčio reikšme lygia 0,02, yra 8,43%.

2.14 pav. Santykinės paklaidos $\sigma_2^*(K^*)$ priklausomybė nuo slenksčio, $N = 1000$

2.14 pav. mažiausia santykinė paklaida, su antrąja paklaidų matavimo metrika, gaunama slenksčio reikšmei lygiai 0,03, yra 6,50%.

Apibendrinant abiejuose grafikuose pateikiamus rezultatus, galime sakyti, kad esant dideliam imties tūriui užtenka mažos slenksčio reikšmės, lyginant su mažesnės imties tūrio imama slenksčio reikšme, kad būtų gaunama mažiausia santykinė paklaida.

2.1 lentelė

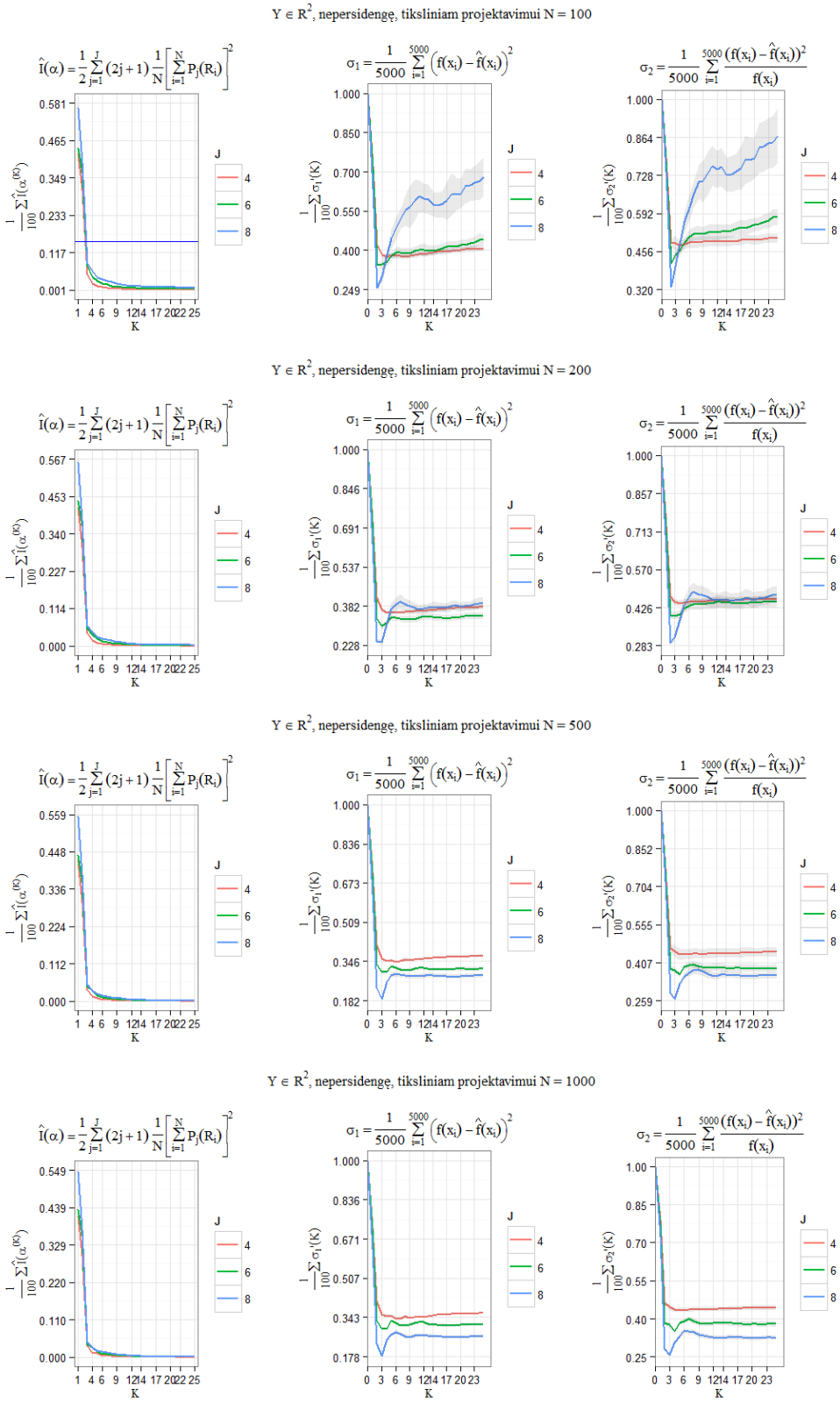
Tankio vertinimo paklaidos naudojant projektavimo indekso reikšmes

Imties tūris	100	200	500	1000
Naudojama 1-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,15	0,045	0,035	0,02
Santykinės paklaidos reikšmė, %	13,94	8,37	6,41	8,43
Naudojama 2-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,15	0,1	0,04	0,03
Santykinės paklaidos reikšmė, %	18,86	9,08	5,82	6,50

Iš 2.1 lentelės matome, kad esant mažam imties tūriui, šiuo atveju $N = 100$, santykinė paklaida yra mažiausia su slenksčio reikšme lygia 0,15 abejais tankio vertinimo atvejais. Tačiau procentinės santykinės pirmos paklaidos reikšmės yra mažesnės, negu antros paklaidos reikšmės.

Taip pat galime išskirti atvejį esant dideliam imties tūriui, pavyzdžiui $N = 500$, $N = 1000$. Šiuo atveju slenksčio reikšmės parenkamos gana mažos, lyginant su slenksčio reikšmių parinkimu, kai $N = 100$. Vertinant daugiamačio tankio įverčio tikslumą su pirmąją paklaidų matavimo metrika santykinės paklaidos gaunamos didesnės, nei vertinant tikslumą su antrąja paklaidų matavimo metrika.

Atvejis, kai $N = 200$ įdomus tuo, kad su pirmąja paklaidų matavimo metrika slenksčio reikšmė – 0,045 yra mažesnė už slenksčio reikšmę – 0,1, antrosios paklaidų matavimo metrikos atveju, mažiausiai santykinei paklaidai rasti.



2.15 pav. Projektavimo indekso ir paklaidų grafikas R^2 nepersidengusių klasių atveju, kai $N = 100$ parinkta slenksčio reikšmė $0,15$

2.15 pav. pirmojoje eilutėje vaizduojamas slenksčio parinkimas R^2 nepersidengusių klasių atveju naudojant gautą slenksčio reikšmę iš 2.7 – 2.8 pav. grafikų - 0,15, kai $N = 100$.

Iš 2.15 pav. pateikiamų paklaidų grafikų (antrasis ir trečiasis grafikų stulpeliai) matome, kad kryptyse mažesnėse už optimalų krypčių kiekį K^*, kurioje gaunama minimali tankio vertinimo paklaida, tankio vertinimo paklaidos yra didesnės, nei projektavimo kryptyse esančiose už optimalaus krypčių kiekio K^*. Tai reiškia, kad parinkus per mažą krypčių kiekį daugiamačiam tankiui vertinti, gausime didesnes santykines paklaidas, lyginant su didesniu pasirinktu projektavimo krypčių kiekiu.

2.15 pav. pirmosios eilutės antrajame ir trečiajame grafikuose vaizduojamos tankio vertinimo paklaidos naudojant abejas paklaidų matavimo metrikas. Pagal pirmąją paveikslo dalį, kurioje vaizduojamos projektavimo indekso reikšmės su 0,15 parinkta slenksčio reikšme, matome, kad optimalus krypčių kiekis K^* ir pagal pirmąją ir pagal antrąją paklaidų matavimo metrikas, su kuriuo gaunama minimali paklaida, lygus dviem. Taigi, kad ir kokį krypties kiekį, esantį į dešinę nuo K^* besirinktume, duomenų projekcija toje kryptyje bus labai artima Gauso skirstinio tankiui. Gaunamos „nebeįdomios“ kryptys, tokiu atveju galima daryti prielaidą apie algoritmo stabdymą.

Visuose 2.15 pav. pateiktuose grafikuose pilkomis juostomis žymimi pasikliautinieji intervalai su 0,95 pasikliovimo lygmeniu. Nagrinėjant pateiktą paveikslą, matome, kad esant nedideliam imties tūriui $N = 100$ pasikliautinieji intervalai labai platūs, t.y. jame yra „tikroji ieškomos populiacijos požymio apibendrintos charakteristikos reikšmė“. [25] Tačiau didėjant turimos imties tūriui pasikliautinieji intervalai itin susiaurėja, kai $N = 500$ ir $N = 1000$ jie beveik sutampa su atvaizduojamomis tankio vertinimo paklaidų reikšmėmis.

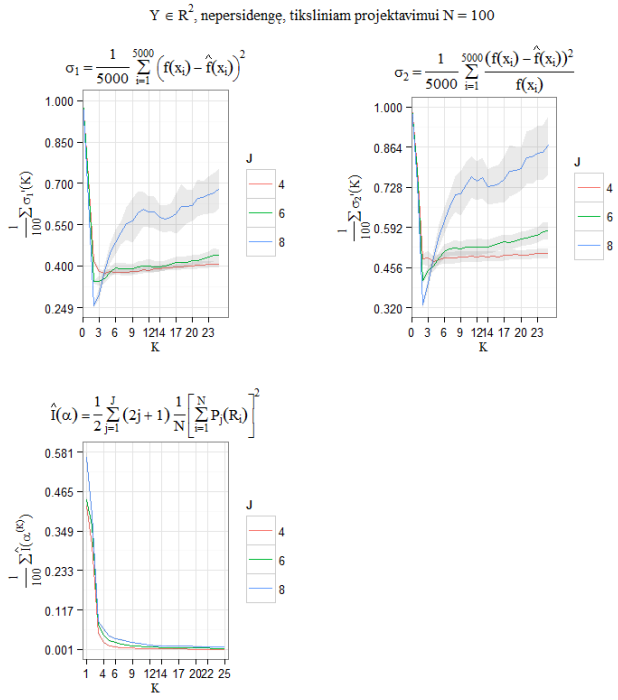
2.3.2. TANKIO ĮVERTINIO TIKSLUMO TYRIMAS, ALGORITMO STABDYMUI, NAUDOJANT NORMALUMO TESTUS

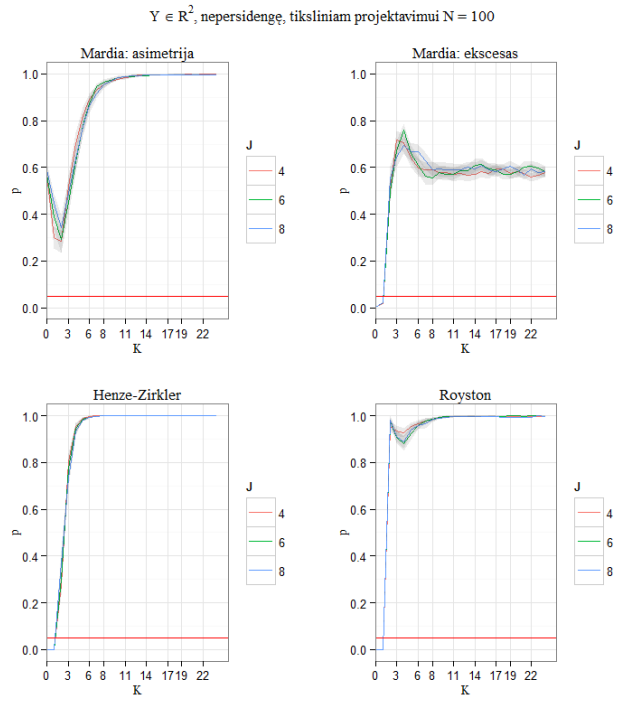
Tyrimui aprašyti naudojami skyriuje **2.3. Tankio įvertinio, pagrįsto tiksliniu projektavimu, tyrimas** nurodyti parametrai. Kaip ir **2.3.1. Tankio įvertinio tikslumo tyrimas, algoritmo stabdymui, naudojant projektavimo indeksą** skyriuje, taip ir naudojant duomenų normalumo testus, siekiame parinkti optimalų projektavimo krypčių kiekį, kad galėtume įvertinti kada būtų palankiausia atlikti algoritmo stabdymą. Darome prielaidą, kad suradus optimalų projektavimo krypčių kiekį, sekančios krypties suradimas gaunamo rezultato nepagerina, o duomenų skirstinys yra standartinis Gauso skirstinys. Tam tikslui įvedame naują dydį – slenkstį - tam tikrą duomenų normalumo testų p reikšmės lygmenį, kuriuo vertiname ar p reikšmė K-tojoje kryptyje yra žemiau lygmens reikšmės, o $K+1$- ojoje kryptyje viršija šį lygmenį. Naudodamiesi slenksčio pasirinkimo galimybe, galime įvertinti, kad nuo surastos K-tosios krypties sekančiose projektavimo kryptyse duomenys bus pasiskirstę pagal standartinį Gauso skirstinį. Projektavimo krypčių kiekį, kuris tenkina p reikšmės slenksčio sąlygą žymime K^*.

Tikslinio projektavimo algoritmu naudojant **(1.40)** ir **(1.41)** išraiškose nurodytas paklaidų matavimo metrikas, daugiamačio skirstinio tankio vertinimui, buvo gautos paklaidos. Kadangi tyrimui naudojome 100 imčių, visos gaunamos paklaidos yra suvidurkintos ir galiausiai nubrėžti paklaidų bei Mardia, Henze – Zirkler, Royston H normalumo testų p reikšmių grafikai, pateikiami 2.16 pav. – 2.19 pav. Likusiems R^2 atvejo persidengimams, bei R^3, R^4, R^5 dimensijoms su visais persidengimo lygmenimis, grafinės iliustracijos pateikiamos 3 priede.

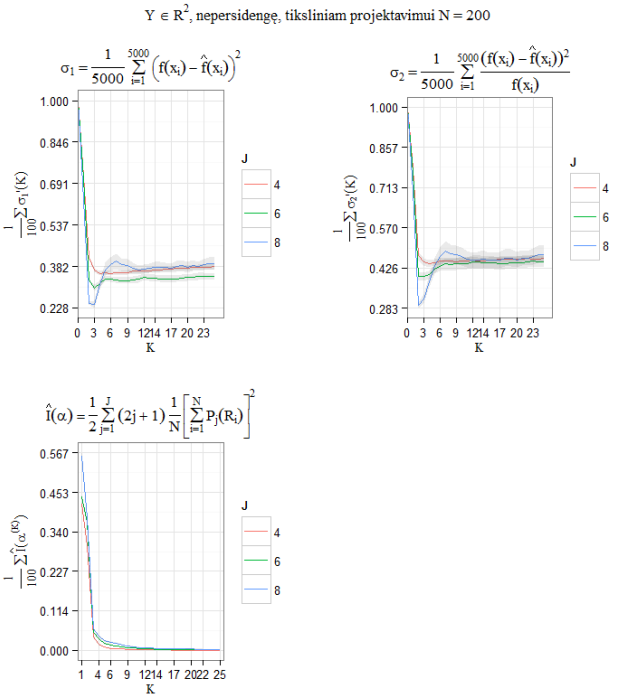
Iš pateiktųjų R^2grafikų matome, kad pagal pirmąją paklaidų vertinimo metriką, kai imties tūris yra mažas $N=100$, o Ležandro polinomų eilė $J=4$ arba $J=6$, daugiamačio tankio įvertinio paklaidas vertinant pagal pirmąją ir antrąją paklaidų matavimo metrikas, matomos nusistovėjusios paklaidų reikšmių tendencijos. Kai $J=8$, paklaidos auga labai sparčiai ir pasikliautinasis intervalas yra labai platus. Esant didesniam imties tūriui paklaidos neturi ryškios tendencijos augti, nepriklausomai nuo Ležandro polinomų eilės J. Duomenų normalumui tikrinti naudojami normalumo testai, nagrinėjant pateikiamus jų grafikus (2.16 pav., 2.17 pav.), kai $N=100, N=200$ po dviejų atliktų duomenų struktūros panaikinimų duomenys yra pasiskirstę pagal standartinį Gauso skirstinį. Didėjant imties tūriui - $N=500$ visi normalumo testai, išskyrus Royston H, rodo (2.18 pav.), kad reikia surasti daugiau nei dvi duomenų projektavimo kryptis. Kai $N=1000$ visais atvejais normalumo testai rodo (2.19 pav.), kad reikia mažiausiai 5-8 krypčių suradimo. Kitų dimensijų atveju paklaidų elgesys yra panašus kaip ir R^2 atveju, nors kartais atsiranda išsiskiriančių atvejų kaip, kad R^4 atveju esant gausiai persidengusioms klasėms.

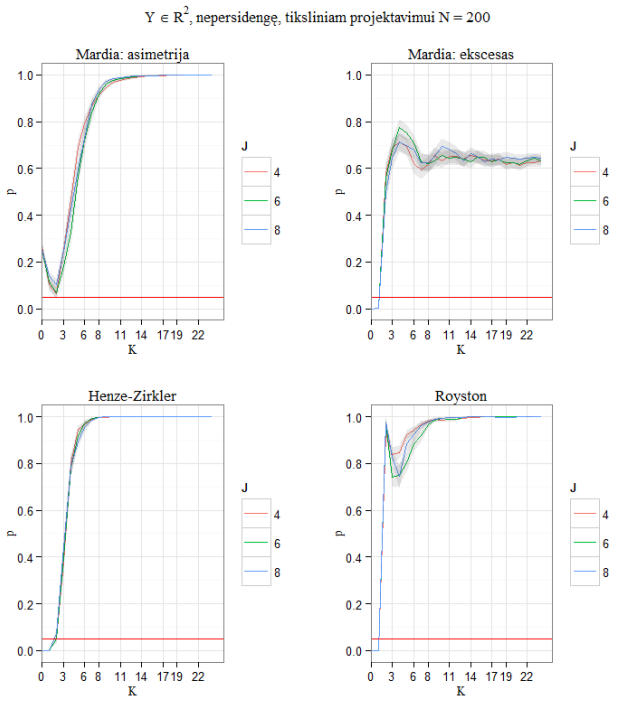
Galutinės išvados bus daromos iš suvidurkintų paklaidų įtraukiant visas tirtas dimensijas.



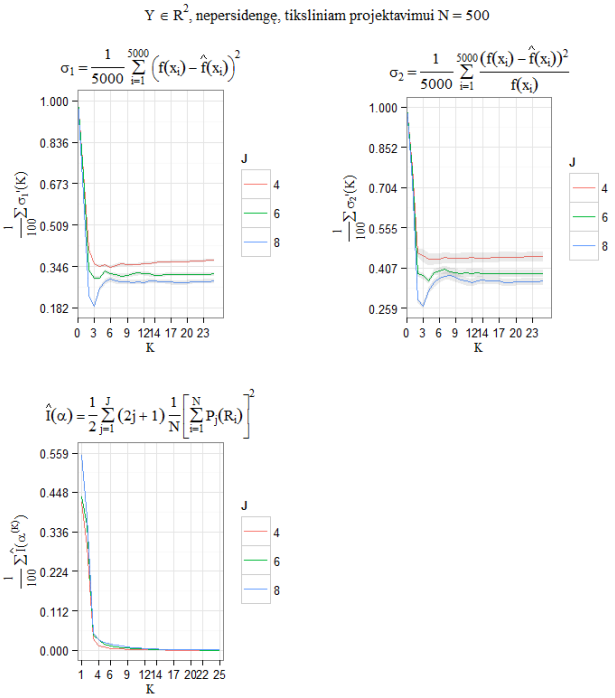


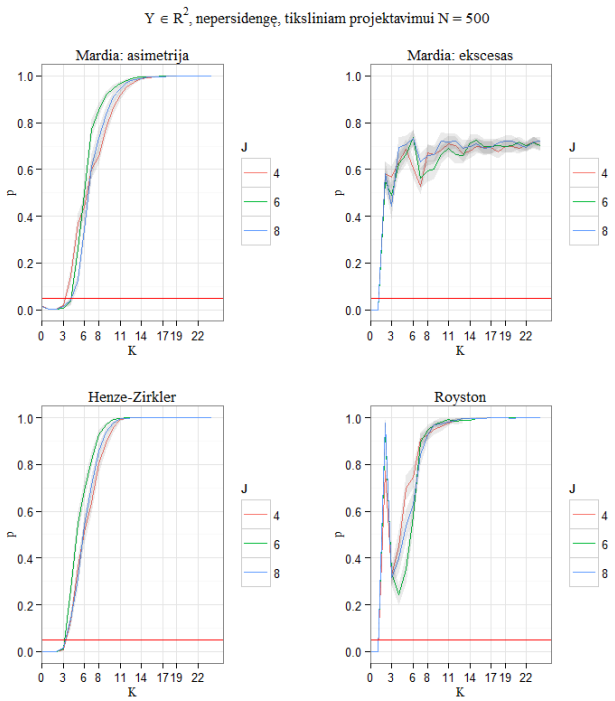
2.16 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 nepersidengusių klasių atveju, kai $N = 100$



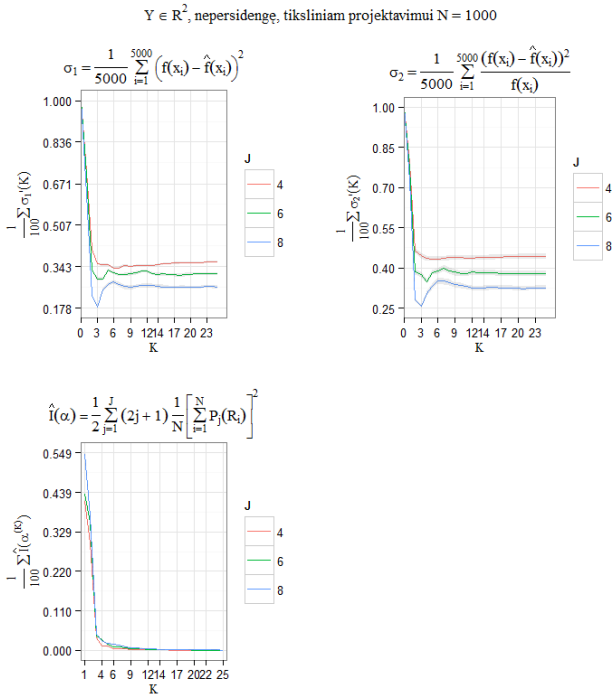


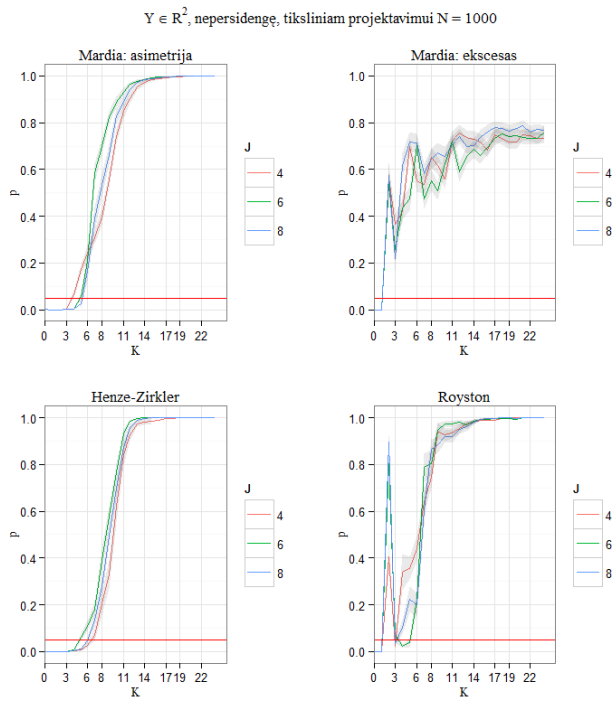
2.17 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 nepersidengusių klasių atveju, kai $N = 200$





2.18 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 nepersidengusių klasių atveju, kai $N = 500$

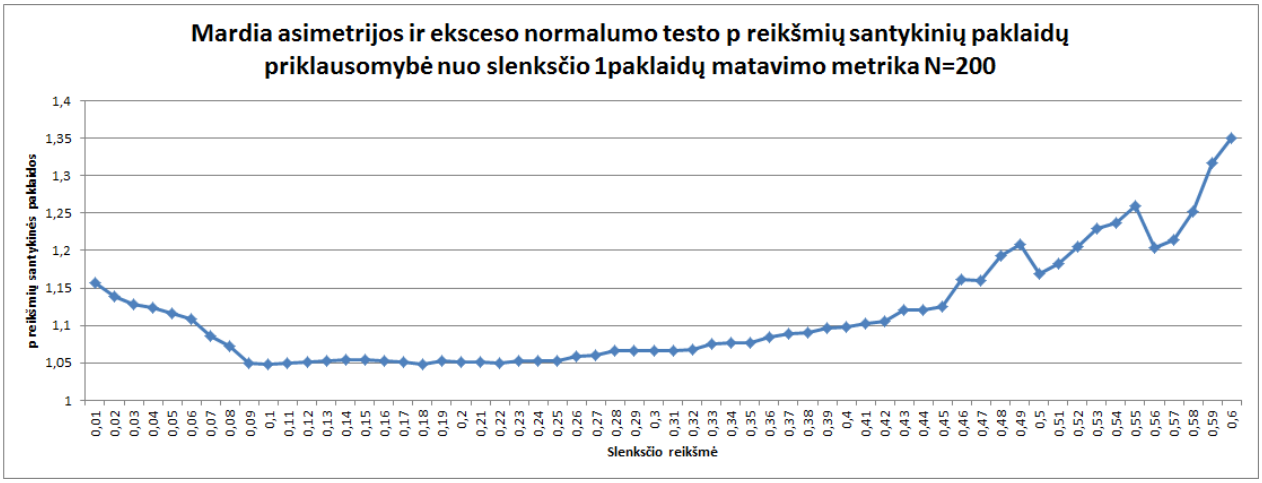




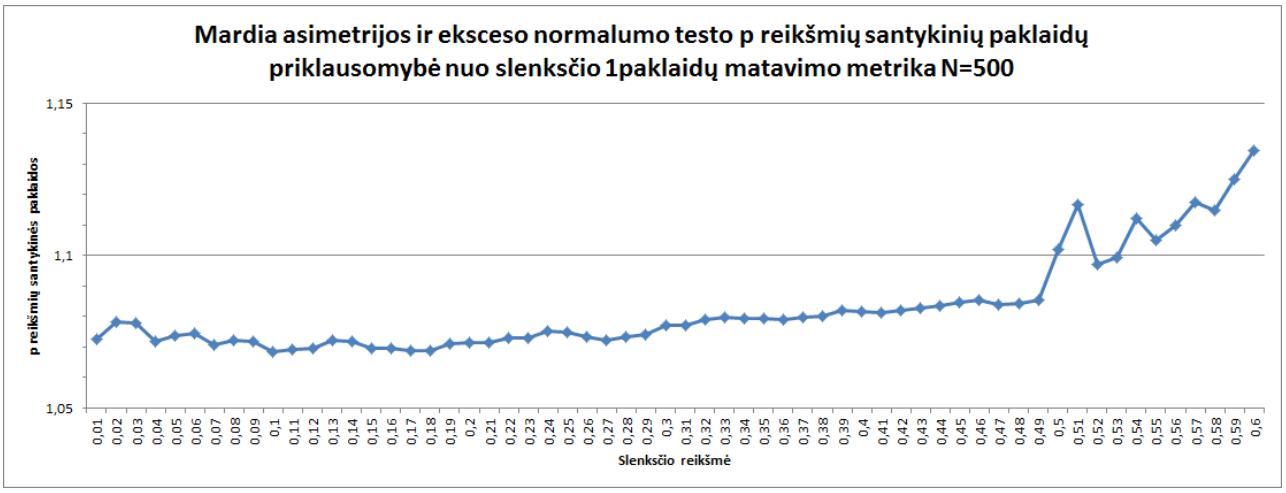
2.19 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 nepersidengusių klasių atveju, kai $N = 1000$

Daugiamačių duomenų normalumo testų p reikšmių santykinių paklaidų nagrinėjimą pradėsime Mardia normalumo testu. 2.20 pav. ir 2.23 pav. pateikiamos grafinės iliustracijos, Mardia normalumo testo p reikšmių santykinių paklaidų priklausomybės nuo slenksčio parinkimo, kai imties tūriai $N = 100, N = 200, N = 500, N = 1000$.

2.20 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 100$



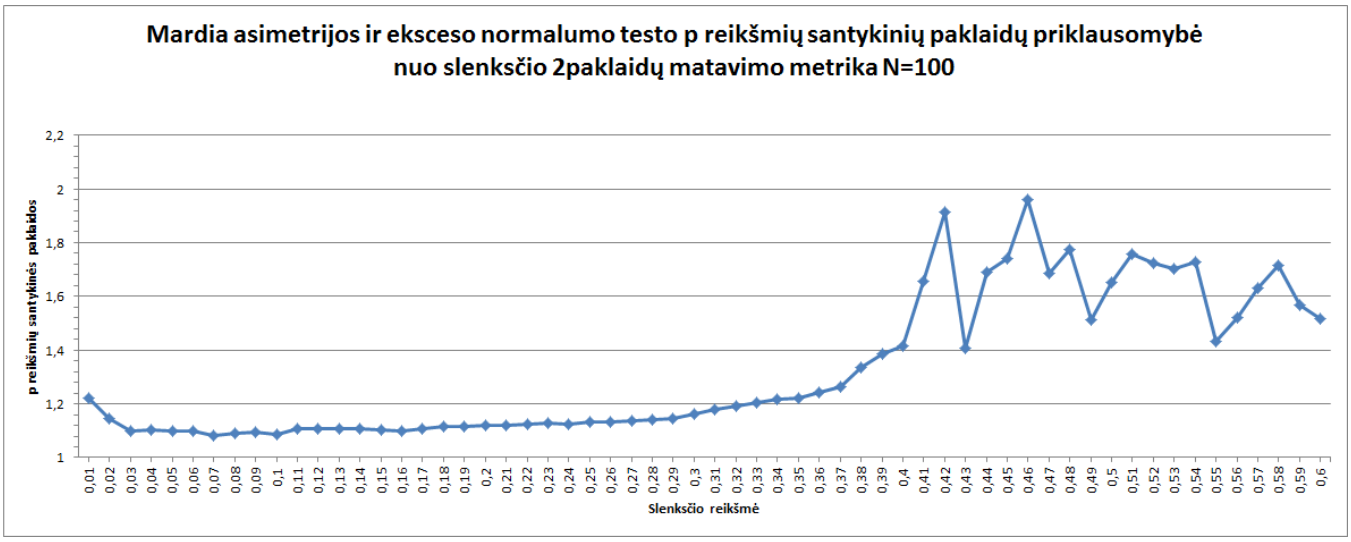
2.21 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 200$



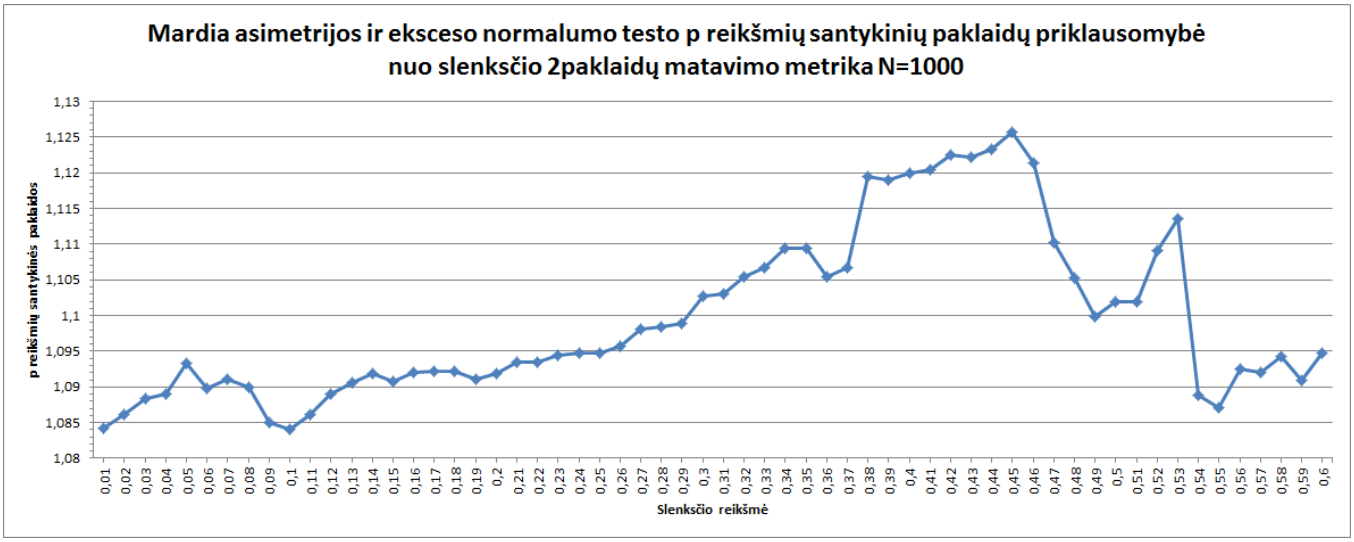
2.22 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas pirmąja metrika $N = 500$

2.23 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas pirmąja metrika $N = 1000$

2.20 - 2.23 paveiksluose matome Mardia normalumo testo, vertinančio daugiamačių asimetrijos ir eksceso p reikšmių, santykinės paklaidos priklausomybę nuo p reikšmės slenkščio parinkimo, gautą naudojant pirmąją paklaidų vertinimo metriką. Pirmuoju atveju, kai $N = 100$ vertinant Mardia normalumo testą pagal asimetriją ir ekscesą kartu, turime imti p reikšmės slenkstį lygų $0{,}29$, kad parinkti krypčių kieki K^* užtikrinantį mažiausią santykinę paklaidą - 5,4%. Kai imties tūris $N = 200$, slenkščio reikšmė – $0{,}1$, o testo p reikšmių santykinė paklaida yra $4{,}79\%$. Imties tūriui $N = 500$, slenkstis – $0{,}1$ ir gaunama santykinė paklaida yra $6{,}84\%$. $N = 1000$ atveju, slenkščio reikšmė – $0{,}21$, o testo p reikšmių santykinė paklaida yra $6{,}63\%$.



2.24 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 100$



2.25 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 1000$

2.24 - 2.25 paveiksluose matome Mardia normalumo testo p reikšmių, santykinės paklaidos priklausomybę nuo p reikšmės slenksčio parinkimo pagal antrąją paklaidų vertinimo metriką. Vertinant Mardia normalumo testą pagal asimetrijos ir eksceso koeficientus kartu, kai $N = 100$ imama slenksčio reikšmė lygi 0,07, o kai imties tūris $N = 1000 - 0{,}1$. Tokiu būdu galime parinkti optimalų krypčių kiekį K^*, užtikrinantį mažiausią santykinę paklaidą. Pirmuoju atveju gauname – 8,36%, antruoju atveju – 8,41%.

Likę Mardia normalumo testo p reikšmių santykinių paklaidų priklausomybės nuo slenksčio parinkimo grafikai, kai $N = 200$, $N = 500$, paklaidas vertinant antrąja metrika, pateikiami 4 priede.

2.2 lentelė

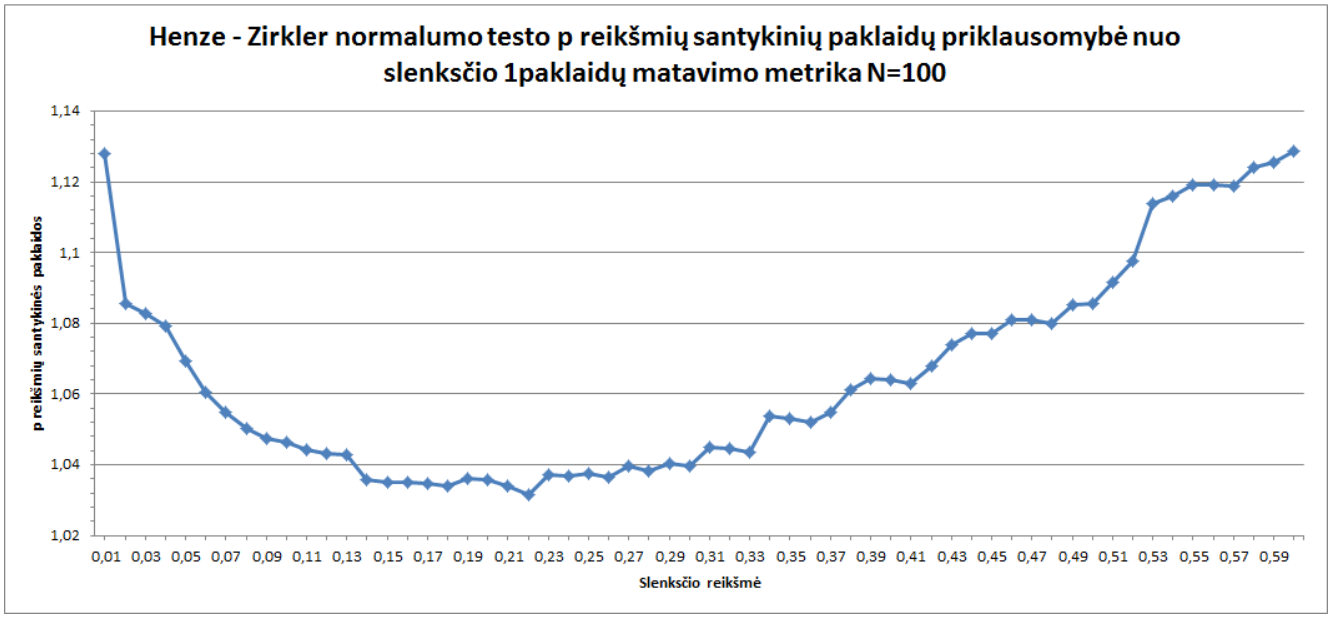
Tankio vertinimo paklaidos naudojant Mardia normalumo testą

Imties tūris	100	200	500	1000
Naudojama 1-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,29	0,1	0,1	0,21
Santykinės paklaidos reikšmė, %	5,4	4,79	6,84	6,63
Naudojama 2-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,07	0,09	0,01	0,1
Santykinės paklaidos reikšmė, %	8,36	9,73	6,79	8,41

Nagrinėjant 2.20 – 2.23 pav. grafiškai ir naudojantis 2.2 lentelėje pateikta rezultatų santrauka pirmajai paklaidų matavimo metrikai, galima pastebėti, kad mažiausia santykinė paklaida gaunama toje srityje, kur išryškėja plokščia kreivės dalis. Todėl galime daryti prielaidą, kad pasirinkus slenksčių intervalą nuo 0,26 iki 0,3 kai $N = 100$, santykinė paklaida mažai nukrypsta nuo optimalios (iki 0,4%). Kai $N = 200$, slenksčių intervalą galime imti nuo 0,09 iki 0,22, jame santykinės paklaidos padidėjimas siekia iki 0,25%; $N = 500$ daugmaž apie 0,5%, kai $N = 1000 - 0.2\%$.

Sakome, jog net ir pasirinkę slenksčius iš pateikiamų intervalų, gausime santykinių paklaidų reikšmes labai artimas gautosioms su optimaliu krypčių kiekiu K^*. Todėl galime teigti, kad matome tendenciją - kuo didesnis imties tūris N tuo santykinės paklaidos reikšmė yra mažiau išsiskirianti iš kitų, t.y. slenksčio parinkimas nėra jautrus.

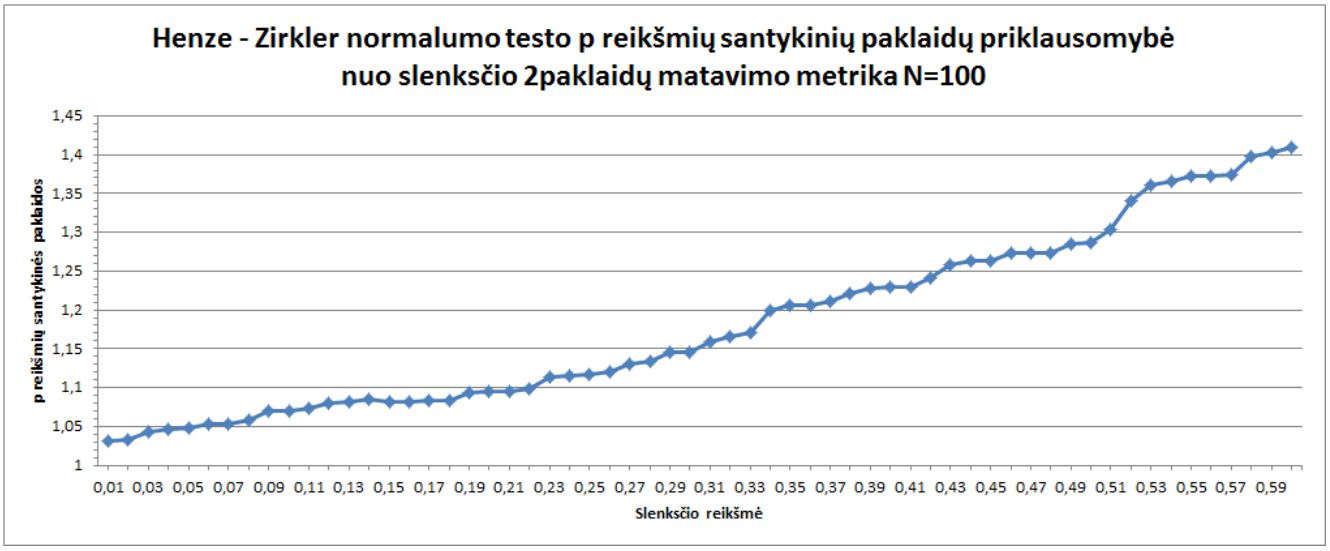
Analogiška tendencija stebima Royston H normalumo testo antrosios paklaidų matavimo metrikos atveju.



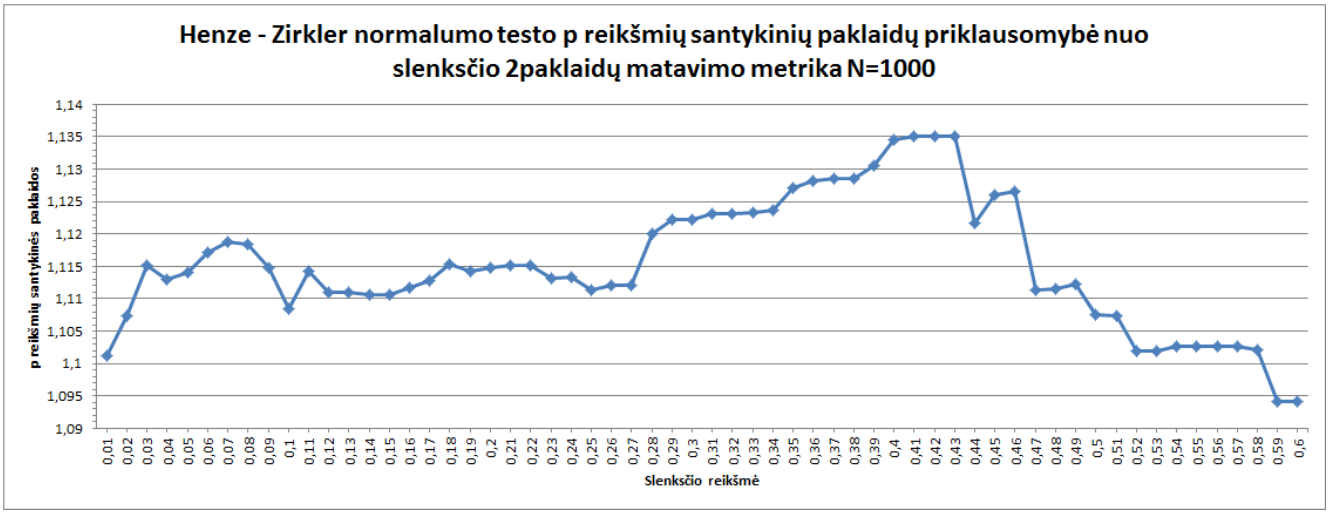
2.26 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 100$

2.27 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 1000$

2.26 - 2.27 paveiksluose matome Henze - Zirkler normalumo testo p reikšmių, santykinės paklaidos priklausomybę nuo p reikšmės slenksčio parinkimo, gautą naudojant pirmąją paklaidų vertinimo metriką. Pirmuoju atveju, kai $N = 100$ turime imti p reikšmės slenkstį lygų $0{,}22$, kad parinkti krypčių kieki K^* užtikrinantį mažiausią santykinę paklaidą – $3{,}13\%$. Kai imties tūris $N = 1000$, slenksčio reikšmė – $0{,}01$, o testo p reikšmių santykinė paklaida yra $8{,}33\%$.



2.28 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 100$



2.29 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 1000$

Iš 2.28 - 2.29 pav. pateiktų grafikų matome, kad Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika, kai imties tūris $N = 100$, imamas p reikšmių slenkstis lygus iki 0,01, o kai $N = 1000 - 0{,}59$. Su tokiais parinktais Henze – Zirkler normalumo testų p reikšmių slenksčiais gaunama mažiausia santykinė paklaida. Pirmuoju atveju – 3,14%, antruoju – 9,42%.

Likę Henze - Zirkler normalumo testo p reikšmių santykinių paklaidų priklausomybės nuo slenksčio parinkimo grafikai, kai $N = 200$, $N = 500$, paklaidas vertinant pirmąja ir antrąja metrikomis, pateikiami 4 priede.

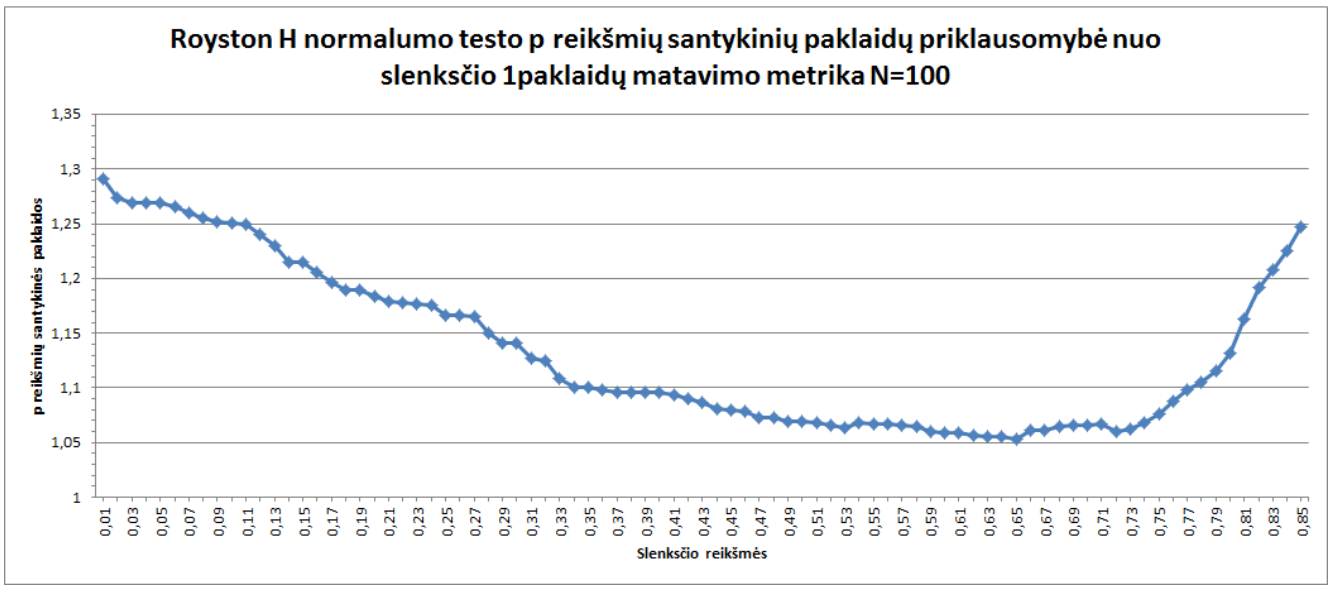
2.3 lentelė

Tankio vertinimo paklaidos naudojant Henze – Zirkler normalumo testą

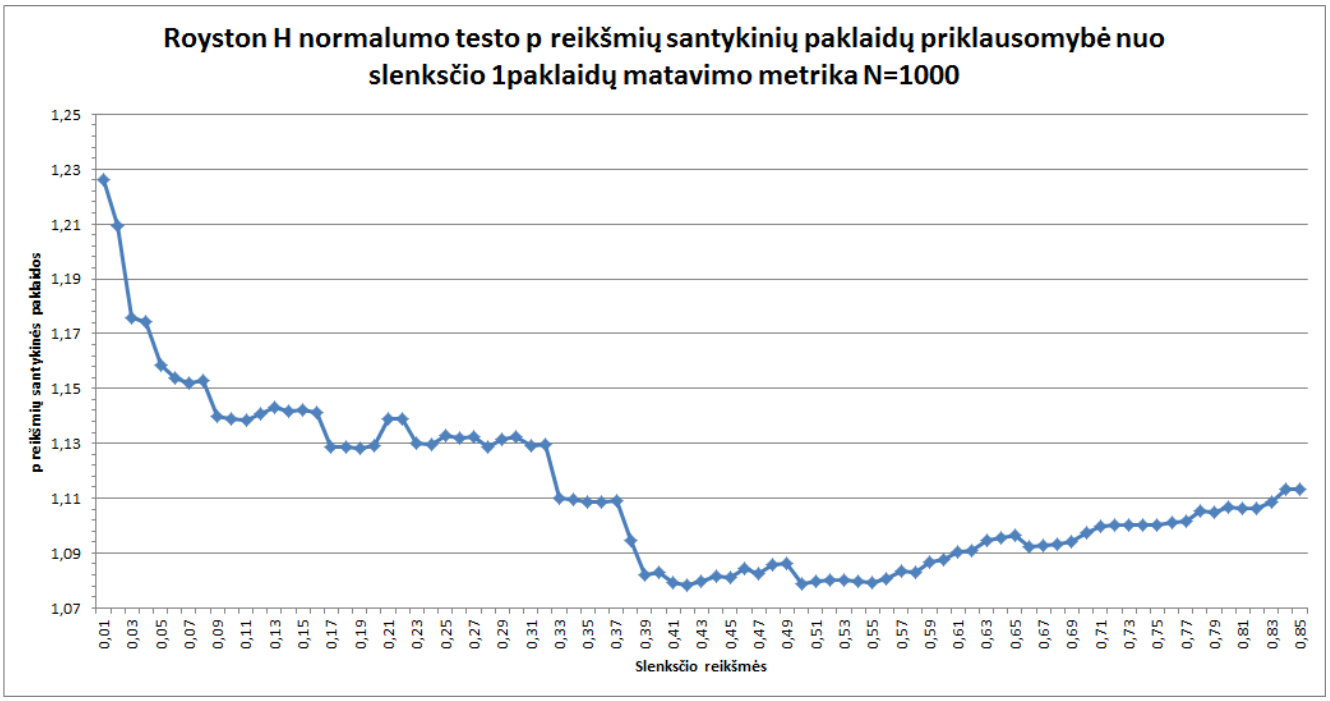
Imties tūris	100	200	500	1000
Naudojama 1-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,22	0,07	0,01	0,01
Santykinės paklaidos reikšmė, %	3,13	3,76	5,14	8,33
Naudojama 2-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,01	0,02	0,01	0,01
Santykinės paklaidos reikšmė, %	3,14	6,99	8,81	10,12

Naudojama pirmoji paklaidų vertinimo metrika Henze – Zirkler normalumo testo atveju. Didėjant imties tūriui, mažėja slenksčio reikšmė, tačiau didėja santykinės paklaidos reikšmė išreikšta procentais.

Pagal antrąją paklaidų vertinimo metriką Henze – Zirkler normalumo testo atveju, didėjant imties tūriui, slenksčio reikšmė išlieka beveik nepakitusi, o santykinės paklaidos reikšmė išreikšta procentais, didėja.



2.30 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 100$



2.31 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 1000$

2.30 - 2.31 paveiksluose matome Royston H normalumo testo p reikšmių, santykinės paklaidos priklausomybę nuo p reikšmės slenksčio parinkimo, gautą naudojant pirmąją paklaidų vertinimo metriką. Pirmuoju atveju, kai $N = 100$ turime imti p reikšmės slenkstį lygų 0,65, kad parinkti krypčių kieki K^* užtikrinantį mažiausią santykinę paklaidą – 5,32%. Kai imties tūris $N = 1000$, slenksčio reikšmė – 0,42, o testo p reikšmių santykinė paklaida yra 7,85%.

2.32 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 100$

2.33 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 1000$

2.32 ir 2.33 paveiksluose matome Royston H normalumo testo p reikšmių, santykinės paklaidos priklausomybę nuo p reikšmės slenksčio parinkimo, gautą naudojant antrąją paklaidų vertinimo metriką. Pirmuoju atveju, kai $N = 100$ turime imti p reikšmės slenkstį lygų 0,33, kad parinkti krypčių kieki K^* užtikrinantį mažiausią santykinę paklaidą – 3,47%. Kai imties tūris $N = 1000$, slenksčio reikšmė – 0,39, o testo p reikšmių santykinė paklaida yra 10,2%.

Likę Royston H normalumo testo p reikšmių santykinių paklaidų priklausomybės nuo slenksčio parinkimo grafikai, kai $N = 200$, $N = 500$, paklaidas vertinant pirmąja ir antrąja metrikomis, pateikiami 4 priede.

2.4 lentelė

Tankio vertinimo paklaidos naudojant Royston H normalumo testą

Imties tūris	100	200	500	1000
Naudojama 1-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,65	0,56	0,5	0,42
Santykinės paklaidos reikšmė, %	5,32	7,14	7,66	7,85
Naudojama 2-oji paklaidų matavimo metrika				
Algoritmo stabdymo slenkstis	0,33	0,13	0,08	0,39
Santykinės paklaidos reikšmė, %	3,47	4,06	7,55	10,2

Royston H normalumo testui su pirmąja paklaidų matavimo metrika, esant įvairiems tūriams N galima optimaliai parinkti slenkstį tam, kad būtų gaunama mažiausia santykinė paklaida.

Su antrąja paklaidų matavimo metrika imties tūriams $(N = 100, N = 200, N = 500)$ galima parinkti slenkstį optimalaus krypčių kiekio radimui, užtikrinančiam mažiausią santykinę paklaidą nuo teoriškai galimos minimalios paklaidos.

Kai imties tūris $N = 1000$, metodas nėra jautrus slenksčio parinkimui, parinkus vienokį ar kitokį, gaunamos labai panašios santykinės paklaidos, jos svyruoja nuo 10% iki 12%.

IŠVADOS

Šiame skyriuje aprašomos gautos darbo išvados galioja keturioms skirtingoms iš eilės einančioms dimensijoms: R^2, R^3, R^4 ir R^5; tūriams N: $100, 200, 500, 1000$ imties taškų; keturioms Gauso skirstinių mišinių klasėms, trims skirtingiems persidengimo lygmenims.

Apibendrinant keturis metodus (pagrįstus projektavimo indeksu, Mardia, Henze – Zirkler ir Royston H normalumo testais), skirtus tikslinio projektavimo algoritmui stabdyti, pateikiame išvadas:

1. Esant mažoms imtims ($N = 100, N = 200$) Henze – Zirkler metodas pateikia geresnius santykinių paklaidų reikšmių rezultatus, lyginant su likusių metodų gaunamais santykinių paklaidų rezultatais;
2. Esant didelėms imtims ($N = 500, N = 1000$) Henze – Zirkler metodas, pagal pirmąją paklaidų matavimo metriką, o projektavimo indeksu pagrįstas metodas, paklaidų vertinimui naudojant antrąją metriką, pateikia geresnius santykinių paklaidų reikšmių rezultatus, lyginant su kitais metodais;
3. Algoritmo stabdymui naudojant metodus pagrįstus normalumo testais, esant dideliam imties tūriui ($N = 1000$) santykinė paklaida yra mažai jautri slenksčio parinkimui;
4. Visose tirtose dimensijose nepersidengusių ir vidutiniškai persidengusių klasių atveju siūloma naudoti Ležandro polinomų eilę $J = 8$; R^2, R^3 dimensijose gausaus persidengimo atveju $J = 6$, o R^4, R^5 - $J = 4$, siekiant gauti tikslesnius daugiamačių tankių įverčius.

Tolimesniuose tyrimuose siūloma detaliau analizuoti Ležandro polinomų eilės parinkimo klausimą. Tikėtina, kad parenkant įvairią Ležandro polinomų eilę skirtingoms projektavimo kryptims, galima gauti tikslesnius daugiamačių tankių įverčius.

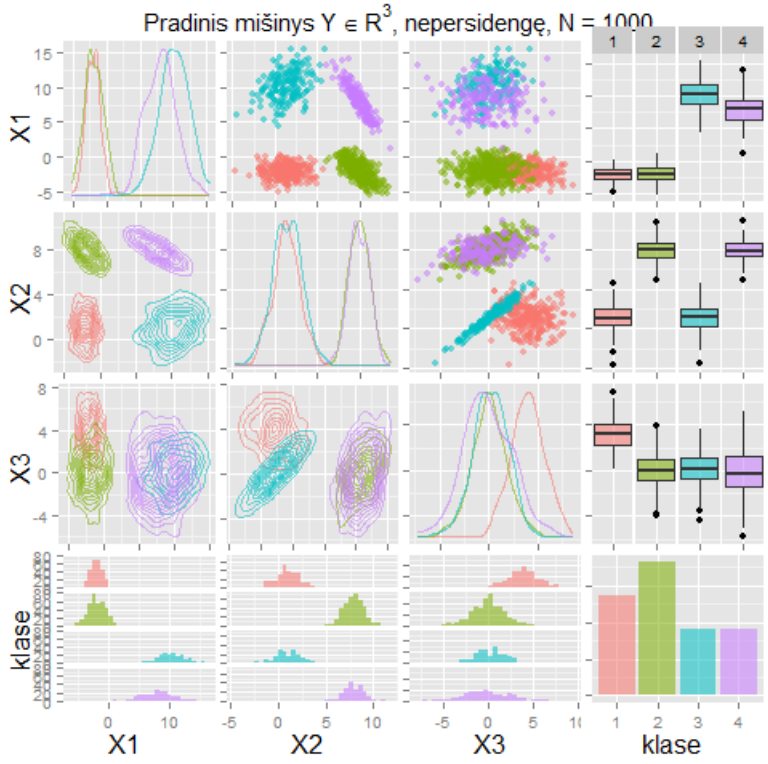
LITERATŪRA

1. Takashi, S., „On the Distribution of Kurstosis Test for Multivariete Normality“, [žiūrėta 2014-04-14]. < http://www.math.sci.hiroshima-u.ac.jp/stat/TR/TR06/TR06-04.pdf>
2. Metodinė priemonė, „ Methods and formulas for normality tests“, [žiūrėta 2014-04-17]. <http://www.stata.com/manuals13/mvmvtestnormality.pdf>.
3. Kankainen, A., Taskinen, S., Oja, H., „On Mardia‘s Tests of Multinormality“, [žiūrėta 2014-04-18]. <http://users.jyu.fi/~slahola/files/icorsmar.pdf>
4. Von Eye, A., Anne Bogat, G., „Testing the assumption of multivariate normality“, Psychology Science, 46, 13p., 2004.
5. Svantesson, T., Wallance, J. W. , „Tests for assessing multivariate normality and the covariance structure of mimo data“, ICASSP (4) : 658-659, 2003.
6. Svantesson, T., Wallance, J. W. , „Tests for assessing multivariate normality and the covariance structure of mimo data“, ICASSP (4) : 656-657, 2003.
7. Korkmaz, S., Goksuluk, D., „Package ‚MVN‘“, metodinė priemonė, [žiūrėta 2014-03-26]. <http://cran.r-project.org/web/packages/MVN/MVN.pdf>.
8. Reynolds, D., „Gaussian Mixture Models“, Encyclopedia of Biometrics , 659-663p., 2009.
9. Royston, J. P., „An extension of Shapiro and Wilk's *W* test for normality to large samples“, Applied Statistics, 31, 119p.
10. Janilionis V., „Koreliacinės ir regresinės analizės pagrindai“, Mokymo kurso medžiaga „Mokymai apie kiekybinių ir kokybinių HSM tyrimų duomenų analizės metodus“, http://www.lidata.eu/index.php?file=files/mokymai/Janilionis_III/jan_III.html&course_file=jan_III_1.html
11. Karbauskaitė R., „Daugiamačių duomenų projekcijos algoritmai, mažinantys atstumų skaičiavimus“, magistro darbas, 9p., . 2005.
12. Dzemyda G., Kurasova O., Žilinskas J., „Daugiamačių duomenų vizualizavimo metodai“, vadovėlis informatikos krypties doktorantams ir magistrantams, 41p., . 2008.
13. Huber, P. J., „ Projection pursuit“, The Annals of Statistics, 13, 436-442p. , 1985.
14. Jones M. C., Sibson, R., „What is Projection Pursuit?“, J. R. Statist. Soc. A, 150, 1-5p. , 1987.
15. Kavaliauskas, M., „Daugiamačių Gauso skirstinių mišinio statistinė analizė, taikant duomenų projektavimą“, daktaro disertacija. 17-20p. , 2005.
16. Friedman J. H., „Exploratory projection pursuit“, J. Amer. Statist. Assoc., 82, 249-266p. , 1987.
17. Friedman J.H., Stuetzle W., Schroeder A., „Projection Pursuit Density Estimation“, Journal of the American Statistical Association, 79 (387), 599–608p. , 1984.

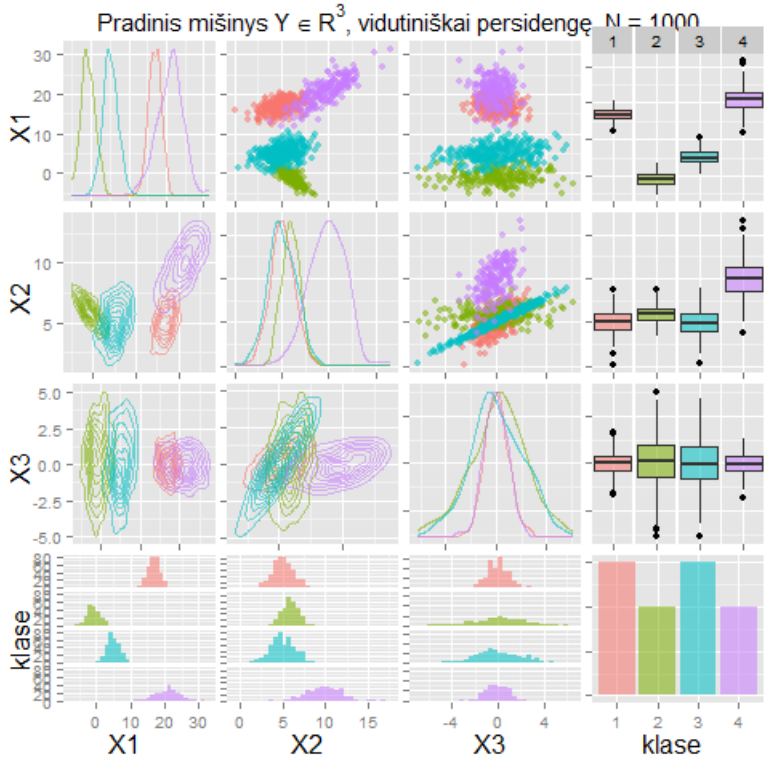
18. Friedman, J. H., Tukey J. W., „A Projection Pursuit Algorithm For Exploratory Data Analysis“, IEEE transactions on computer, c-23, 881-889p. , 1974.
19. Crawford, S. L., Fall, T. C., „Projection pursuit techniques for visualizing high-dimensional data sets“, IEEE Computer Society Press, 94-108p., 1990.
20. Asimov, D., „The grand tour: a tool for viewing multidimensional data“, The SIAM Journal of Scientic and Statistical Computing, Vol. 6, No. 1, 1985.
21. Tukey, P.A., Tukey, J.W, „Graphical display of data sets in 3 or more dimensions“, Interpreting Multivariate Data (V. Barnett, ed.), 89-257p. , 1981.
22. Ward, M. O., LeBlanc, J. T., Tipnis, R. „N-Land: a Graphical Tool for Exploring N-Dimensional Data“, Computer Graphics International Conference in Melbourne, 2-3p. , 1994.
23. Staliulionytė, V. „Tankio vertinimo paklaidos augimo tyrimas naudojant perteklinį projektavimo krypčių kiekį“,/ XII studentų konferencijos pranešimų medžiaga, 37-38p. , 2014.
24. Staliulionytė, V. „Tikslinio projektavimo panaudojimas duomenų vizualizavimui“, XI studentų konferencijos pranešimų medžiaga, 40-41p., 2013.
25. Morkevičius, V. „Statistinė kiekybinių duomenų analizė su SPSS ir STATA“, Mokymų medžiaga. Pavyzdinis metodologinis mokomasis studijų paketas, http://www.lidata.eu/index.php?file=files/mokymai/stat/stat.html&course_file=stat_III_3.html

1 Priedas

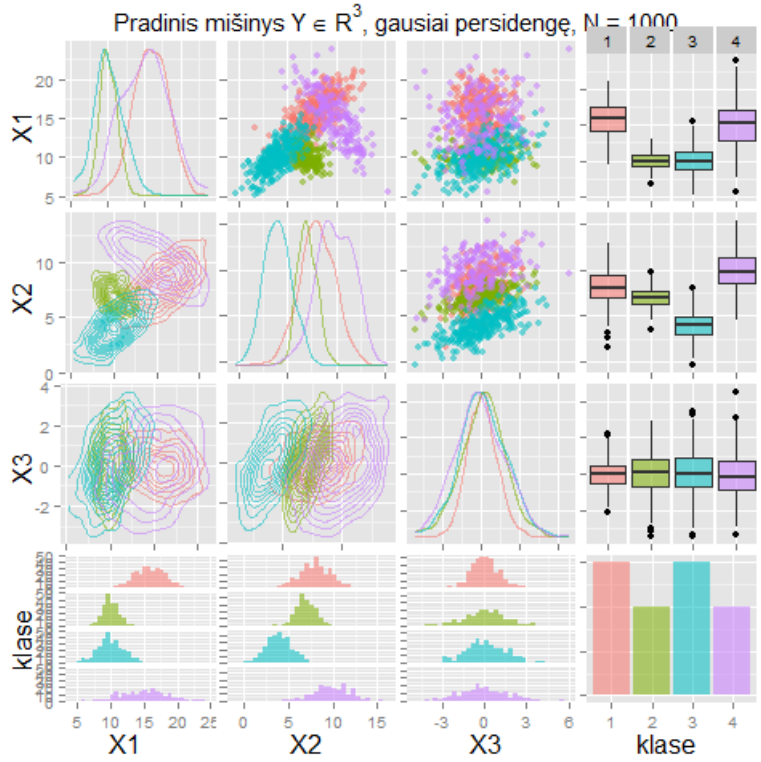
MIŠINIŲ SKIRSTINIŲ PERSIDENGIMO LYGMENYS R^3, R^4, R^5 ATVEJAIS



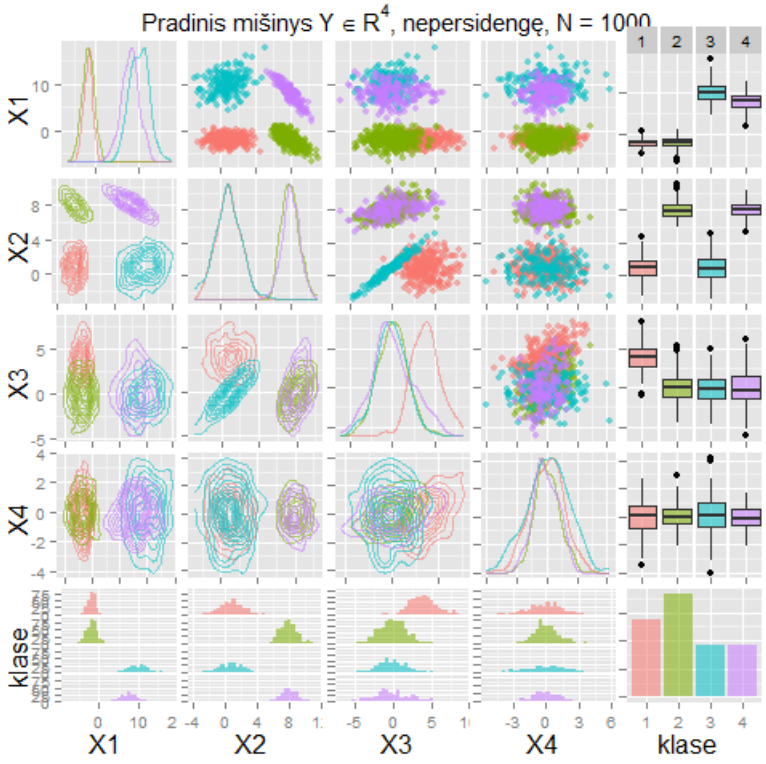
0.1 pav. Nepersidengusių klasių atvejis R^3



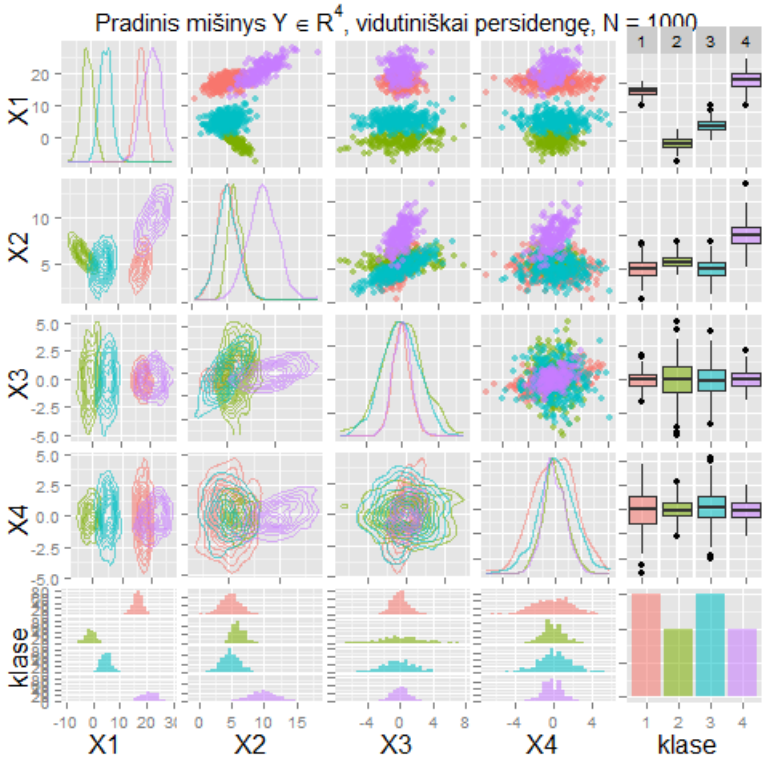
0.2 pav.Vidutiniškai persidengusių klasių atvejis R^3



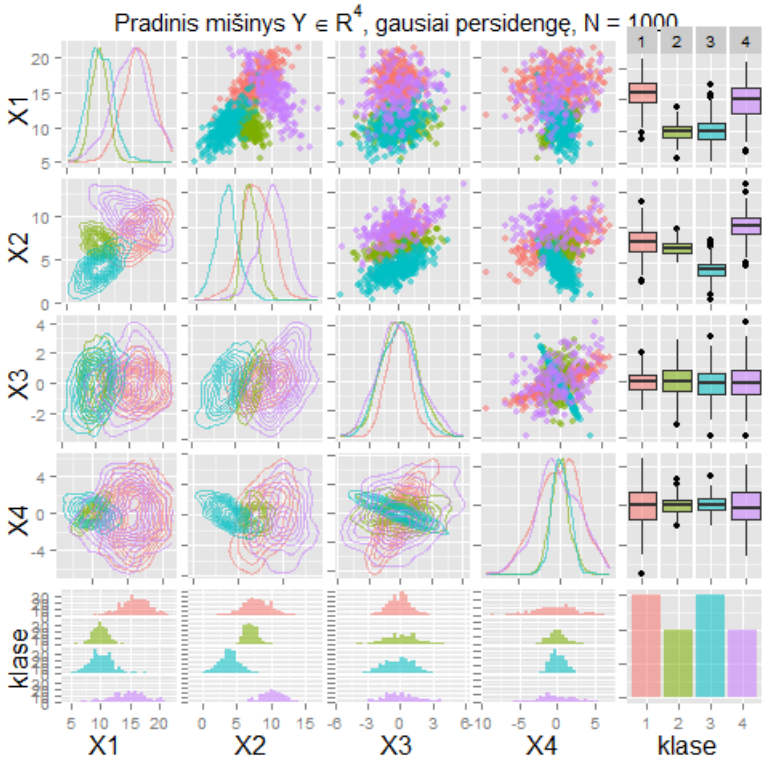
0.3 pav. Gausiai persidengusių klasių atvejis R^3



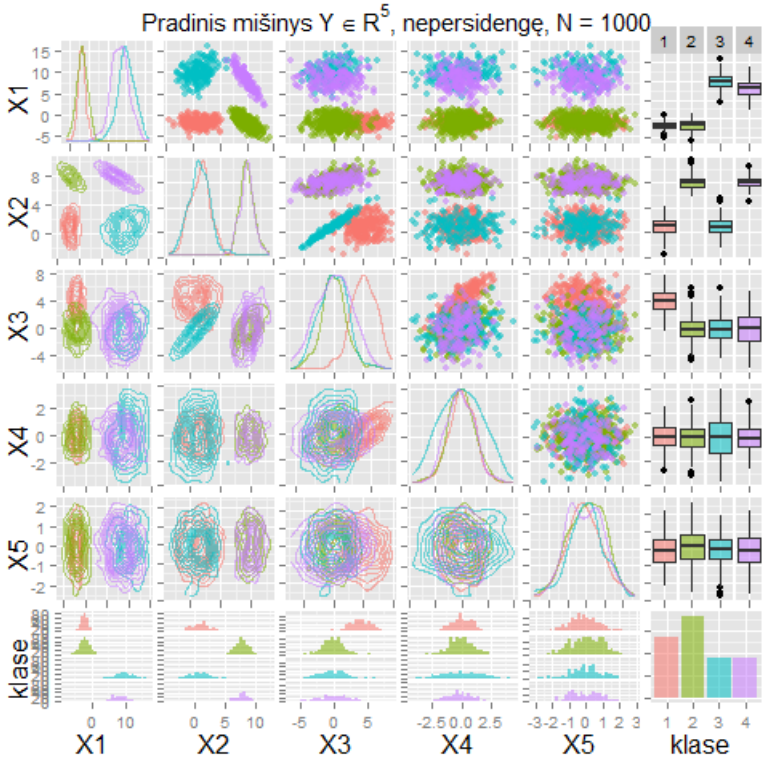
0.4 pav. Nepersidengusių klasių atvejis R^4



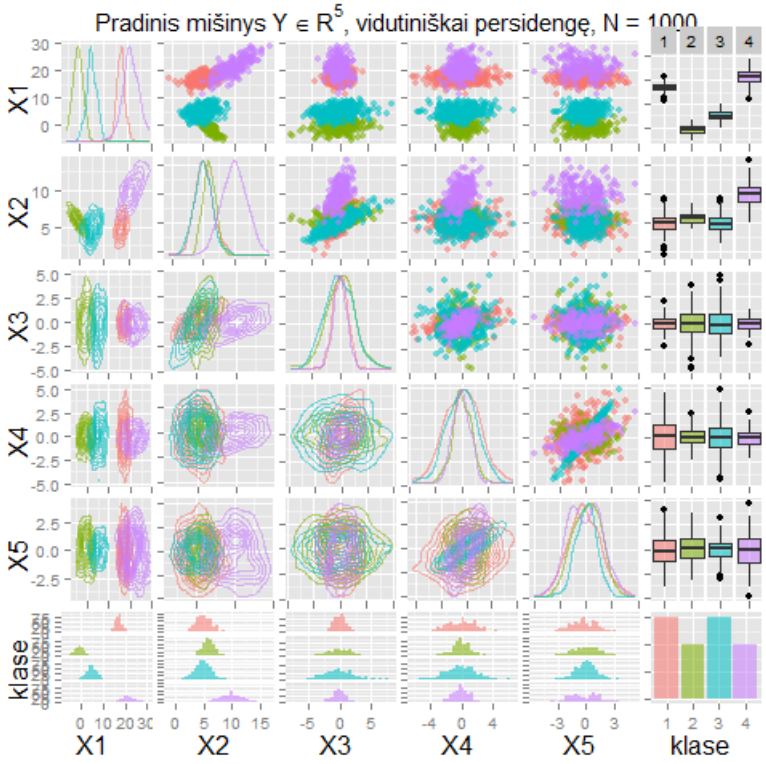
0.5 pav.Vidutiniškai persidengusių klasių atvejis R^4



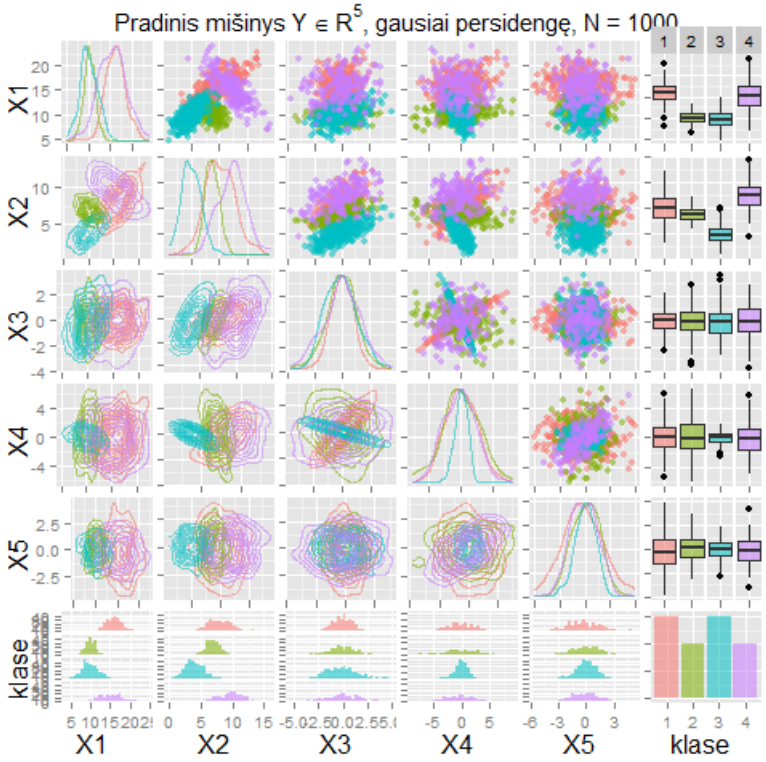
0.6 pav. Gausiai persidengusių klasių atvejis R^4



0.7 pav. Nepersidengusių klasių atvejis R^5



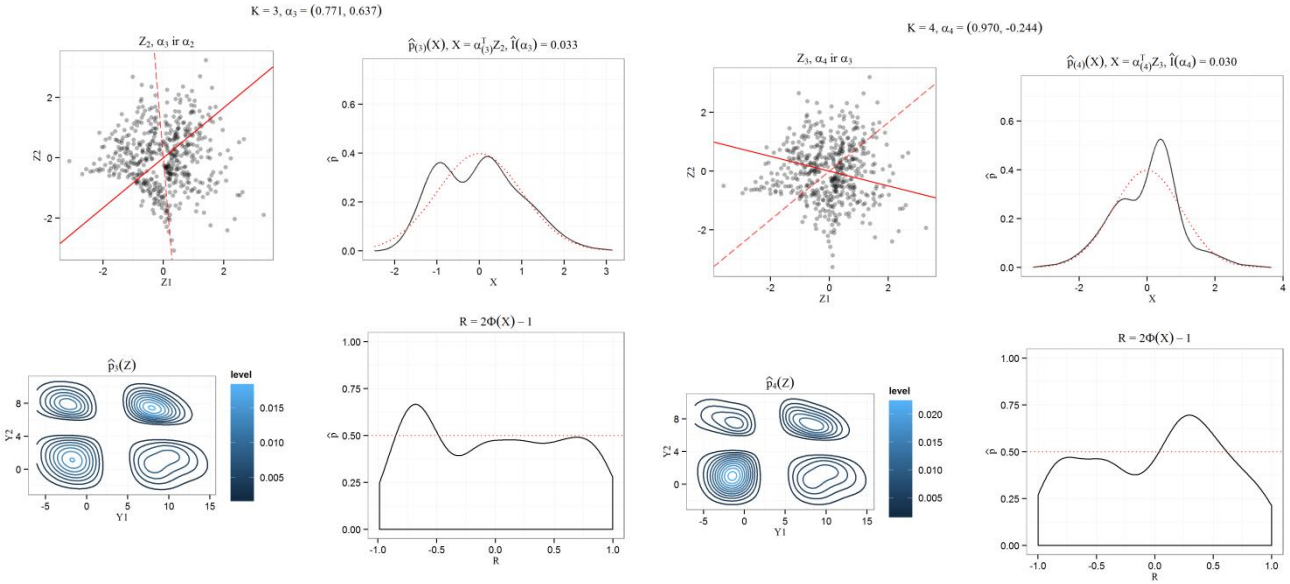
0.8 pav.Vidutiniškai persidengusių klasių atvejis R^5



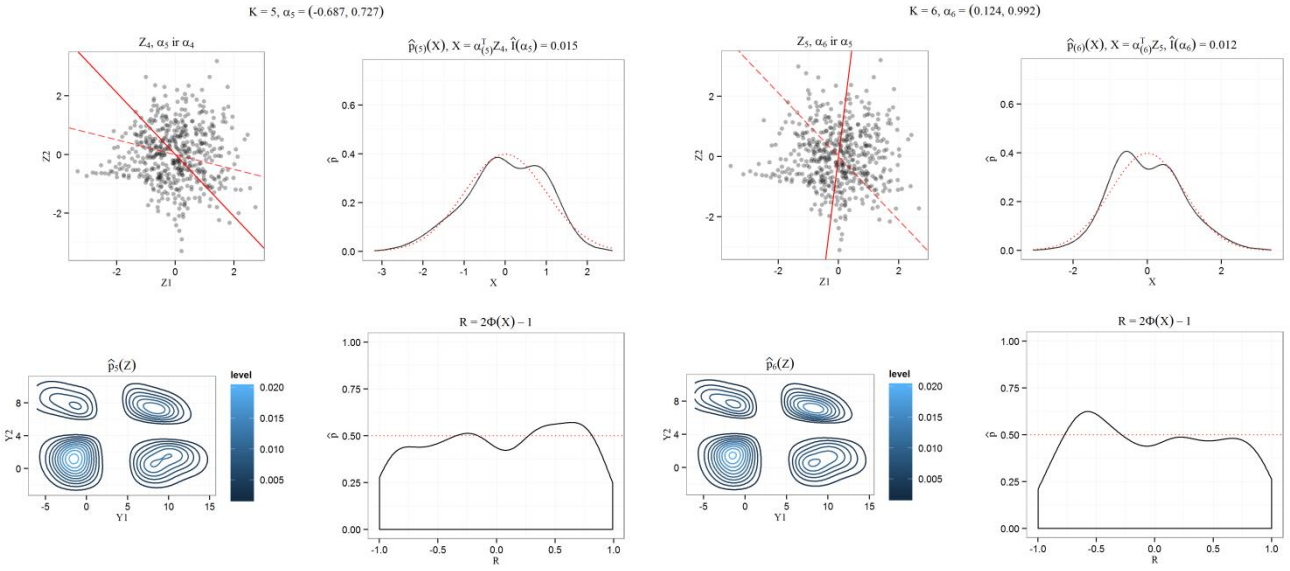
0.9 pav. Gausiai persidengusių klasių atvejis R^5

2 Priedas

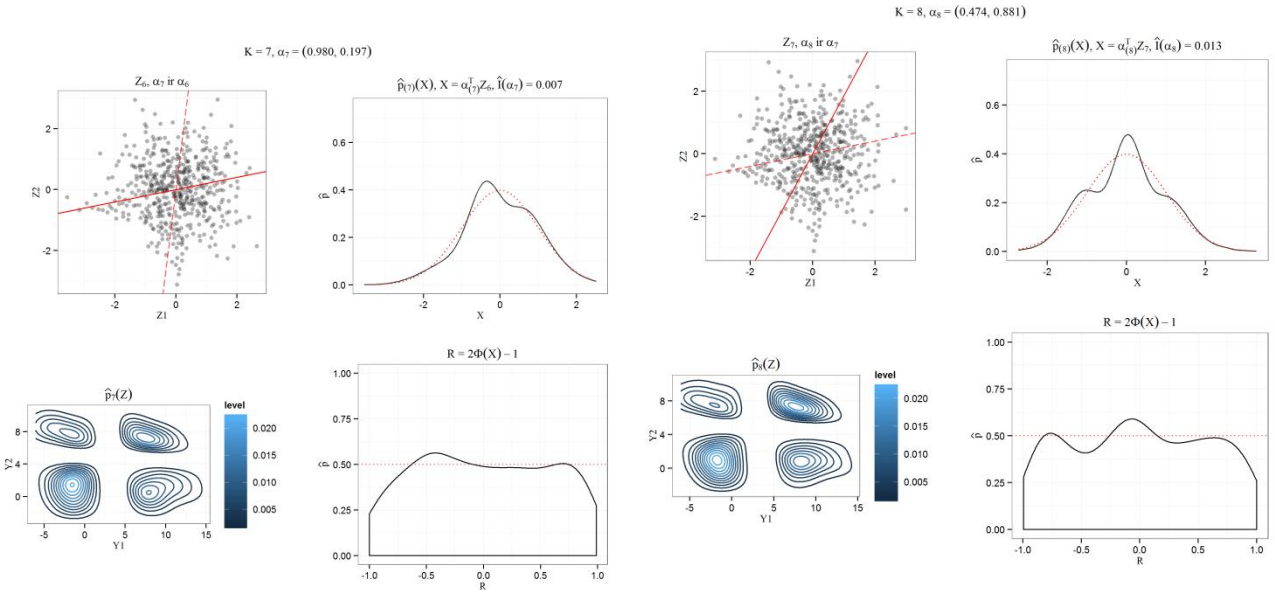
TIKSLINIO PROJEKTAVIMO ALGORITMO ILIUSTRACIJA R^2 ATVEJU



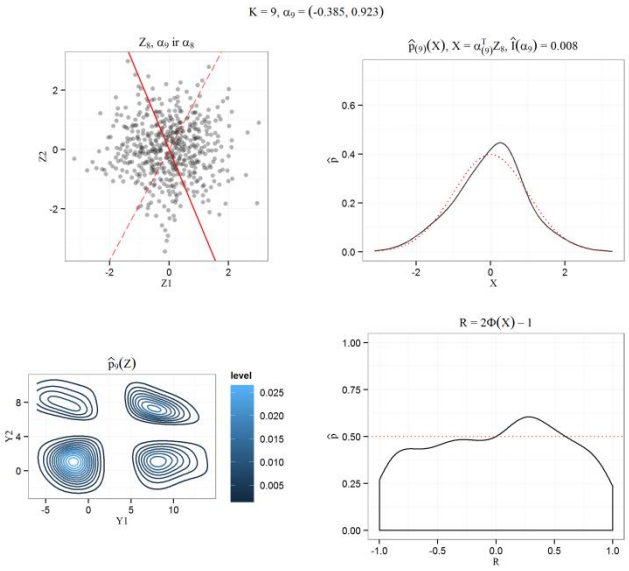
0.10 pav. Sferinta imtis, rastos trečioji ir ketvirtoji duomenų projektavimo kryptys



0.11 pav. Sferinta imtis, rastos penktoji ir šeštoji duomenų projektavimo kryptys



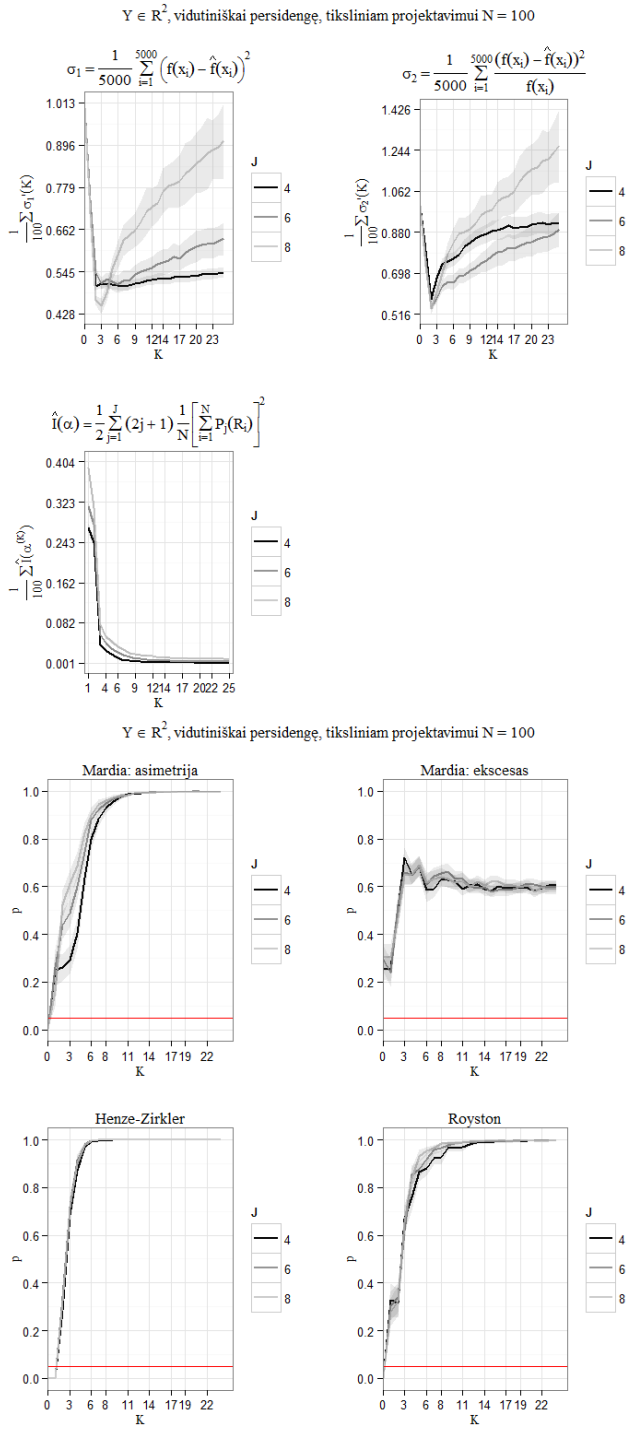
0.12 pav. Sferinta imtis, rastos septintoji ir aštuntoji duomenų projektavimo kryptys



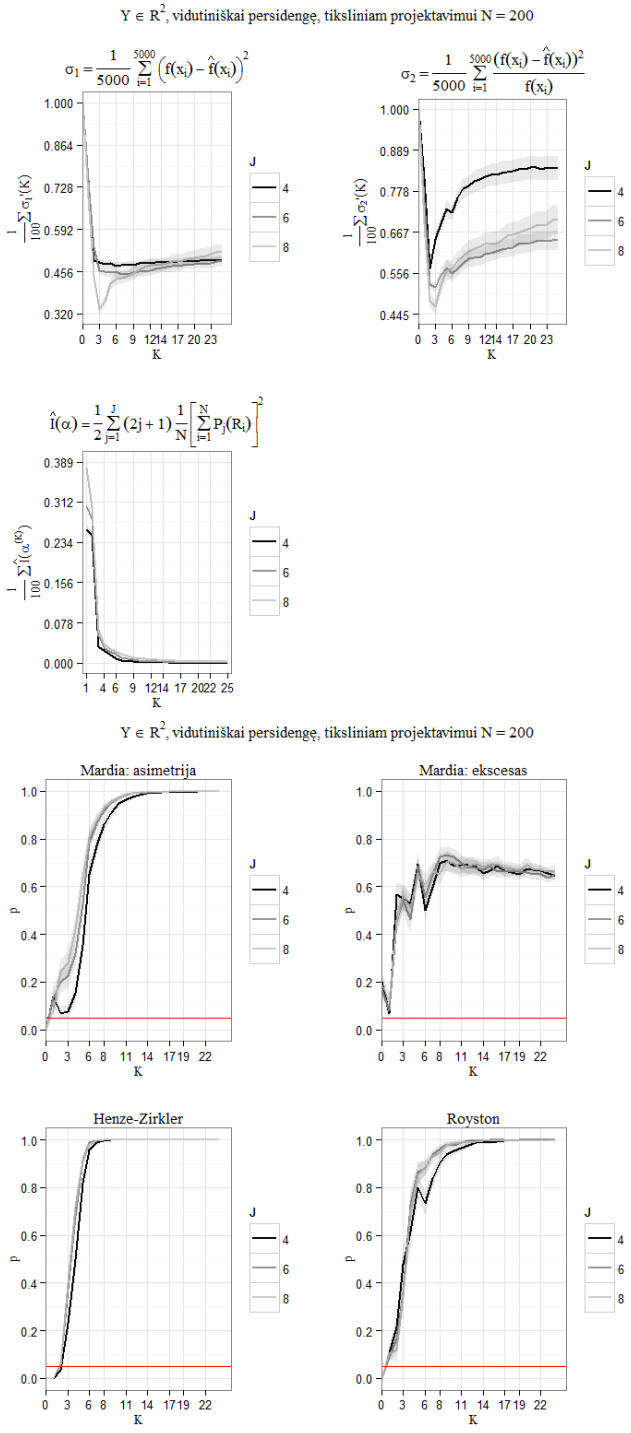
0.13 pav. Sferinta imtis, rasta devintoji duomenų projektavimo kryptis

3 Priedas

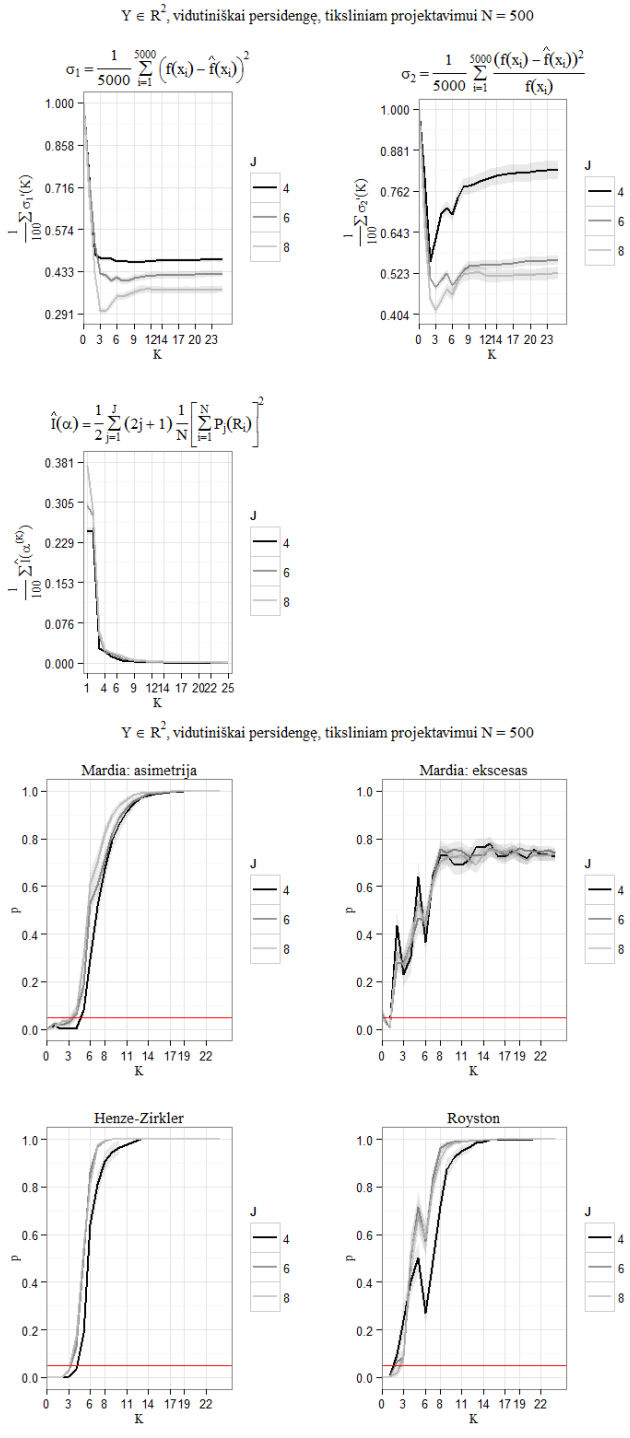
TANKIO ĮVERČIŲ, PAKLAIDŲ, NORMALUMO TESTŲ GRAFIKAI



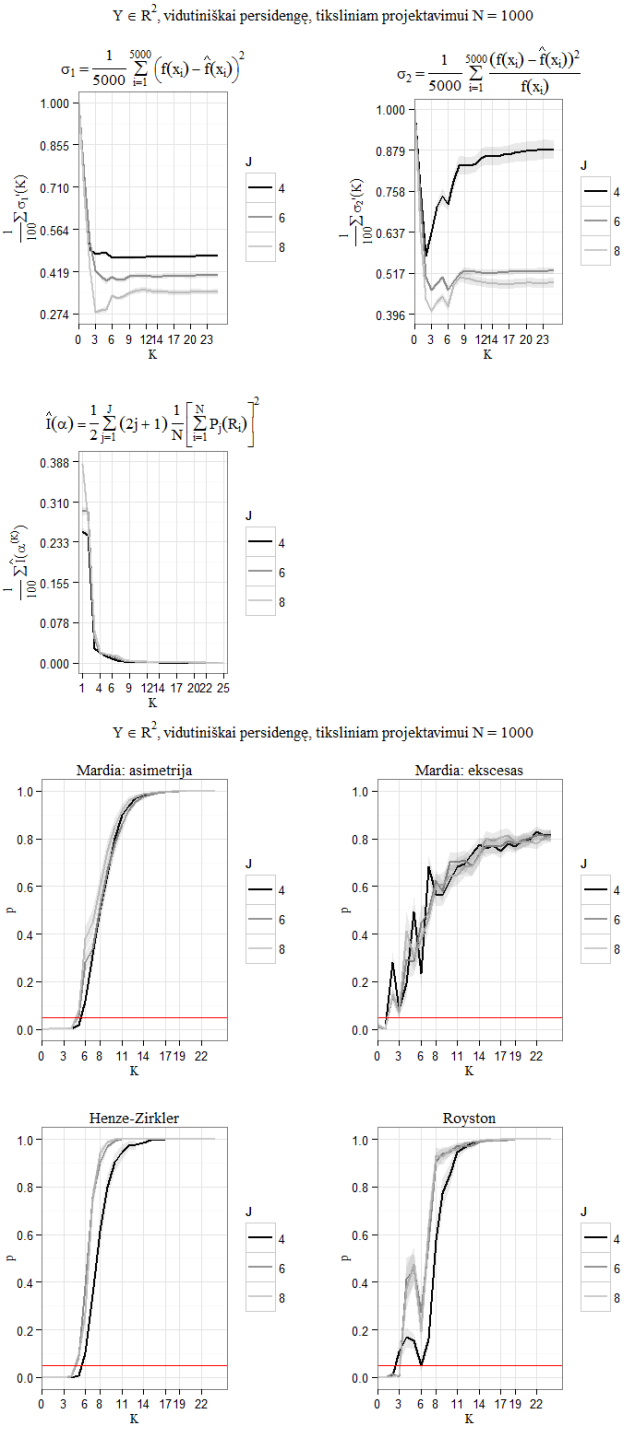
0.14 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 vidutiniškai persidengusių klasių atveju



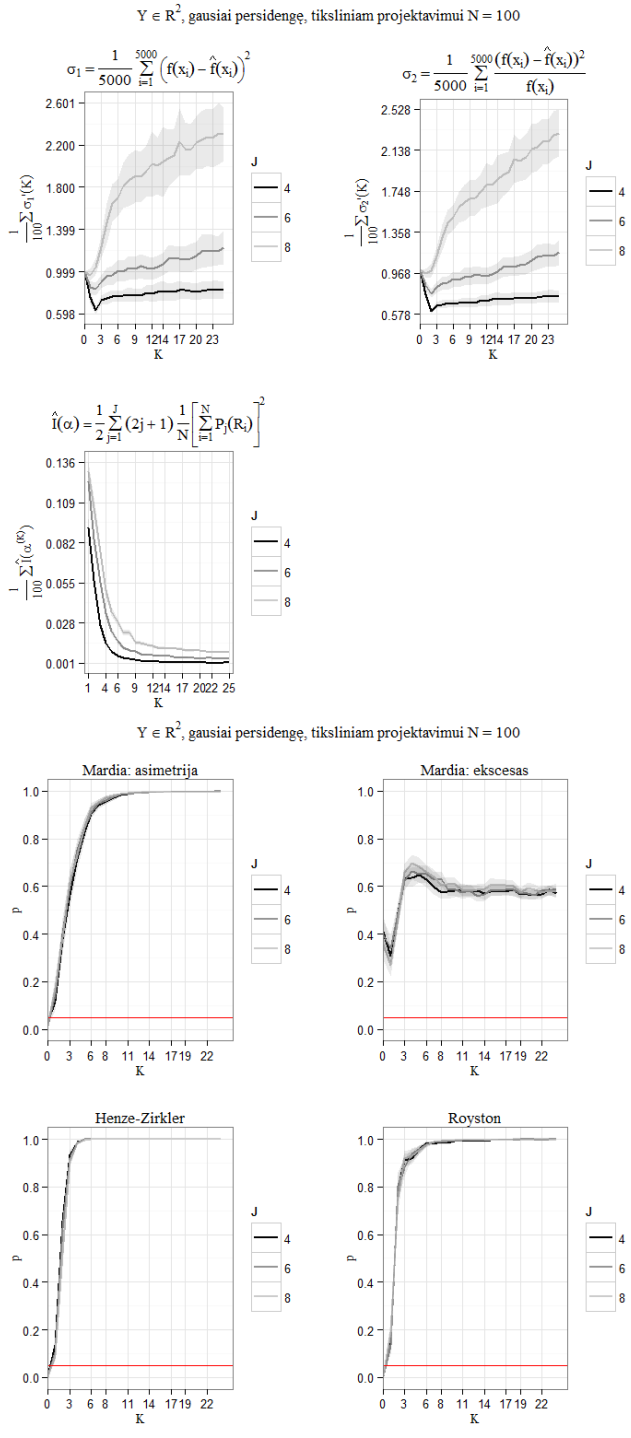
0.15 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 vidutiniškai persidengusių klasių atveju



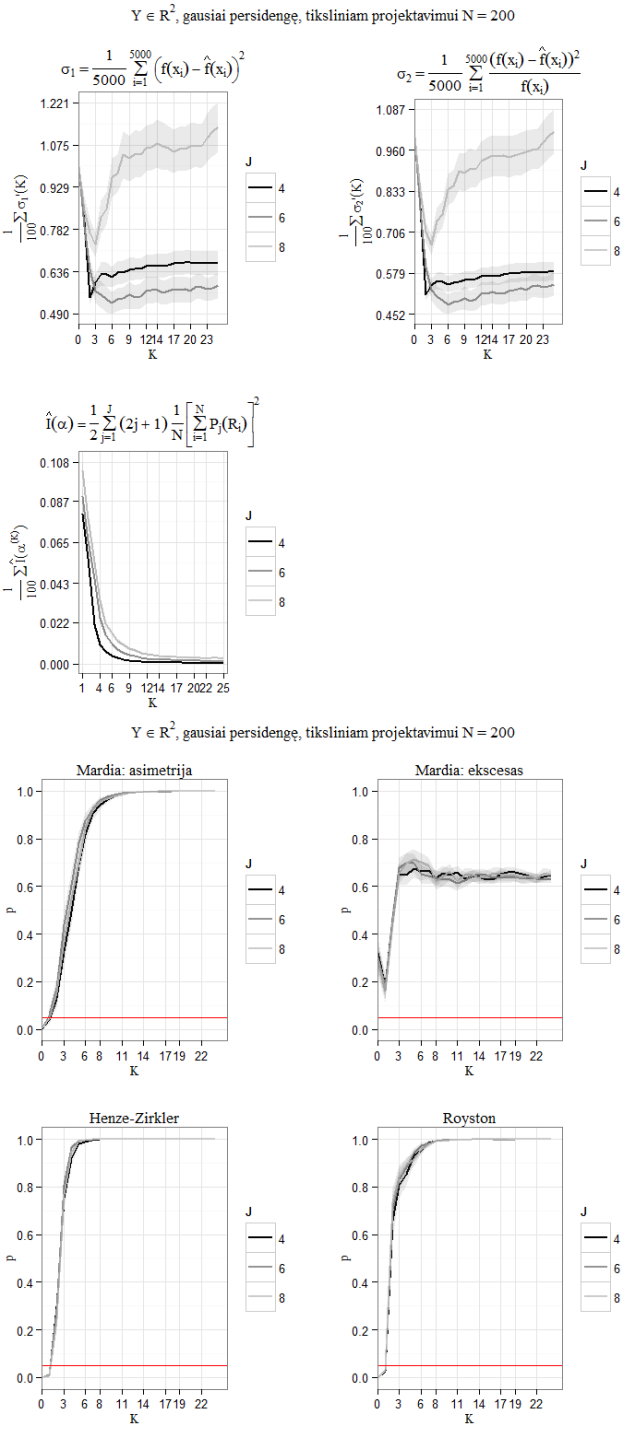
0.16 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 vidutiniškai persidengusių klasių atveju



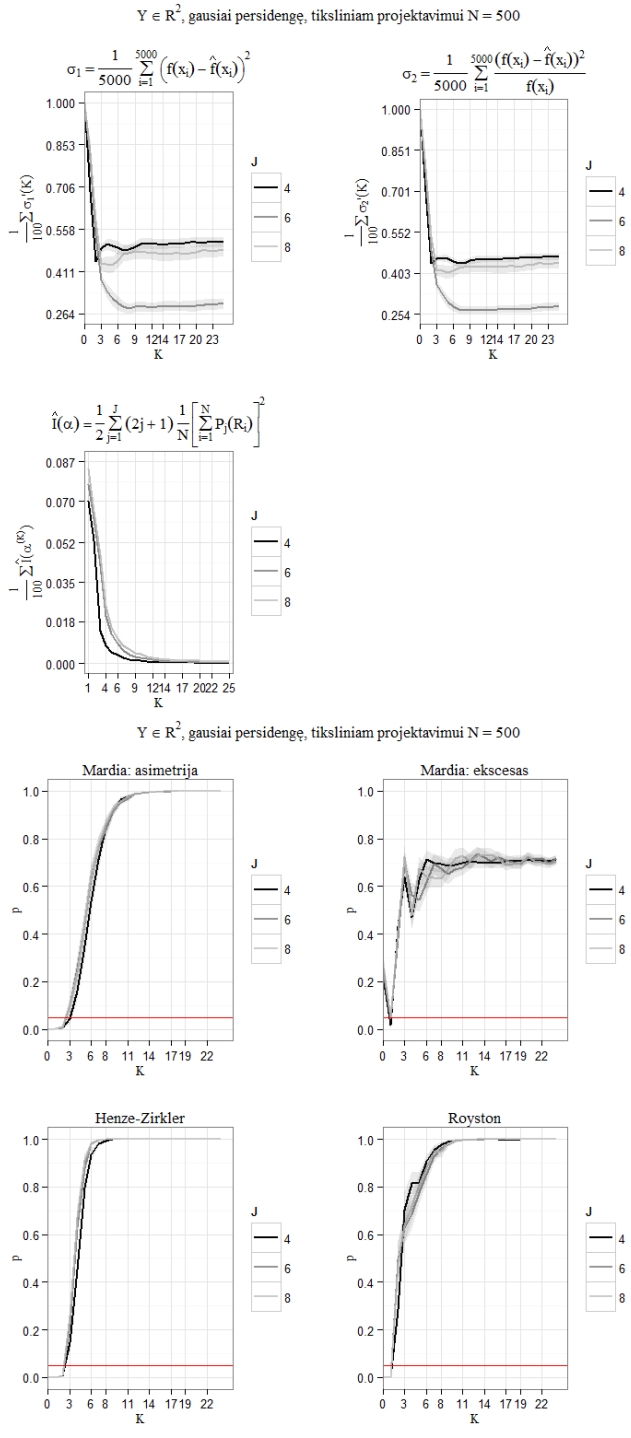
0.17 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 vidutiniškai persidengusių klasių atveju



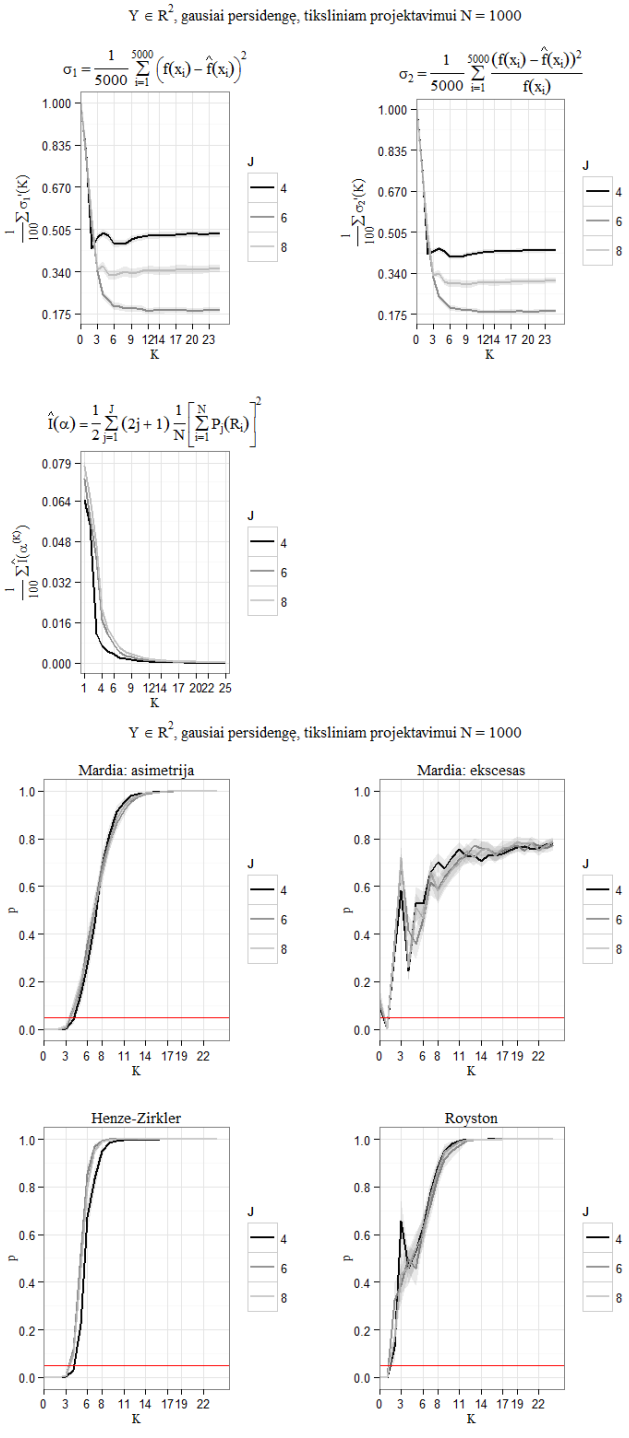
0.18 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 gausiai persidengusių klasių atveju



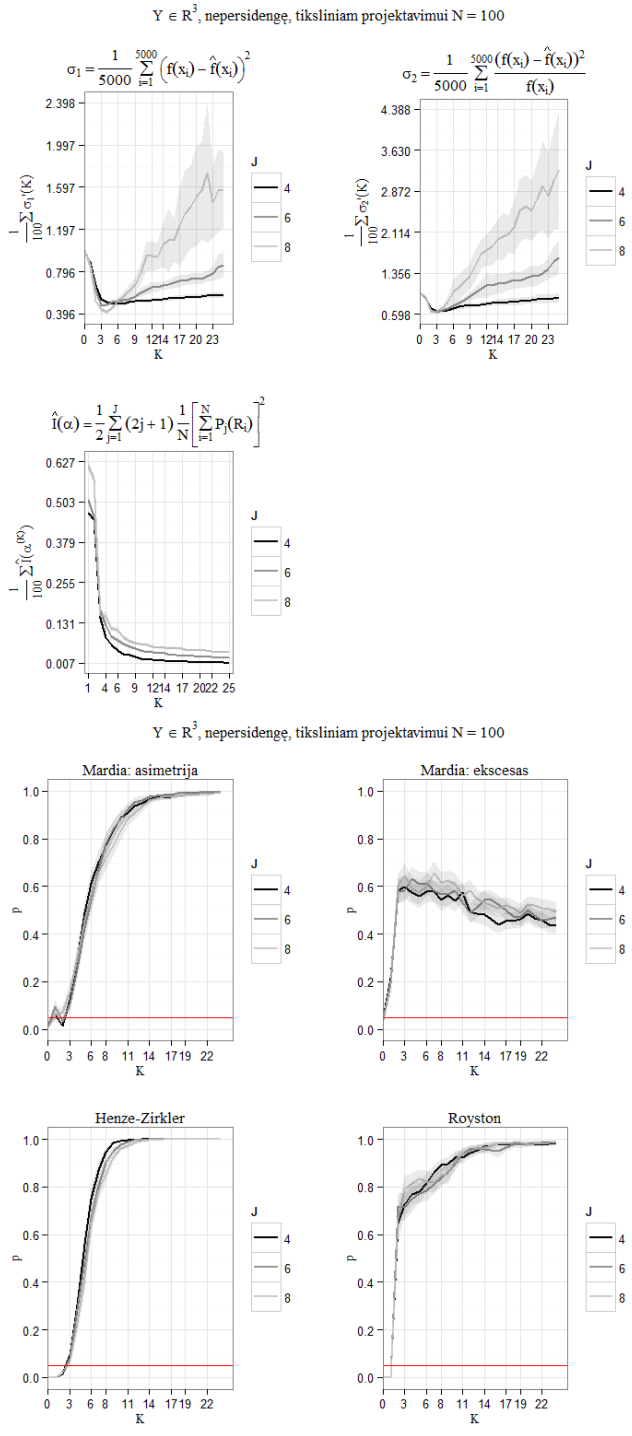
0.19 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 gausiai persidengusių klasių atveju



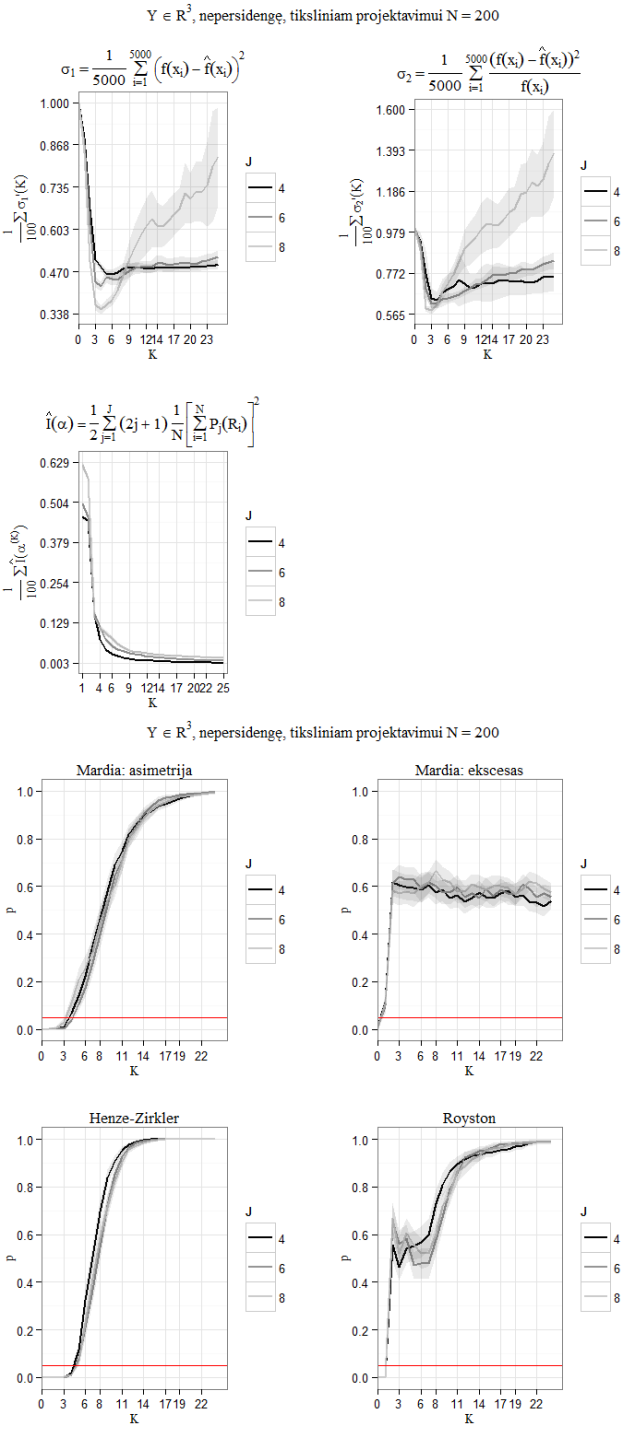
0.20 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 gausiai persidengusių klasių atveju



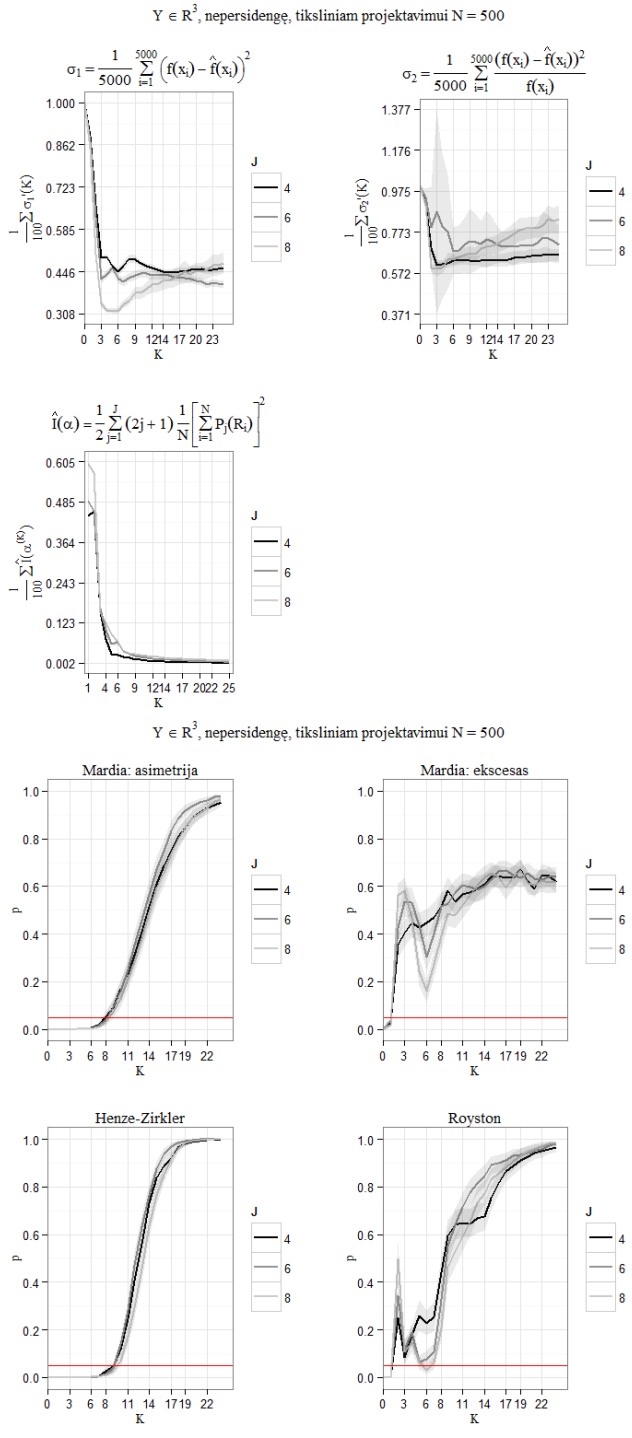
0.21 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^2 gausiai persidengusių klasių atveju



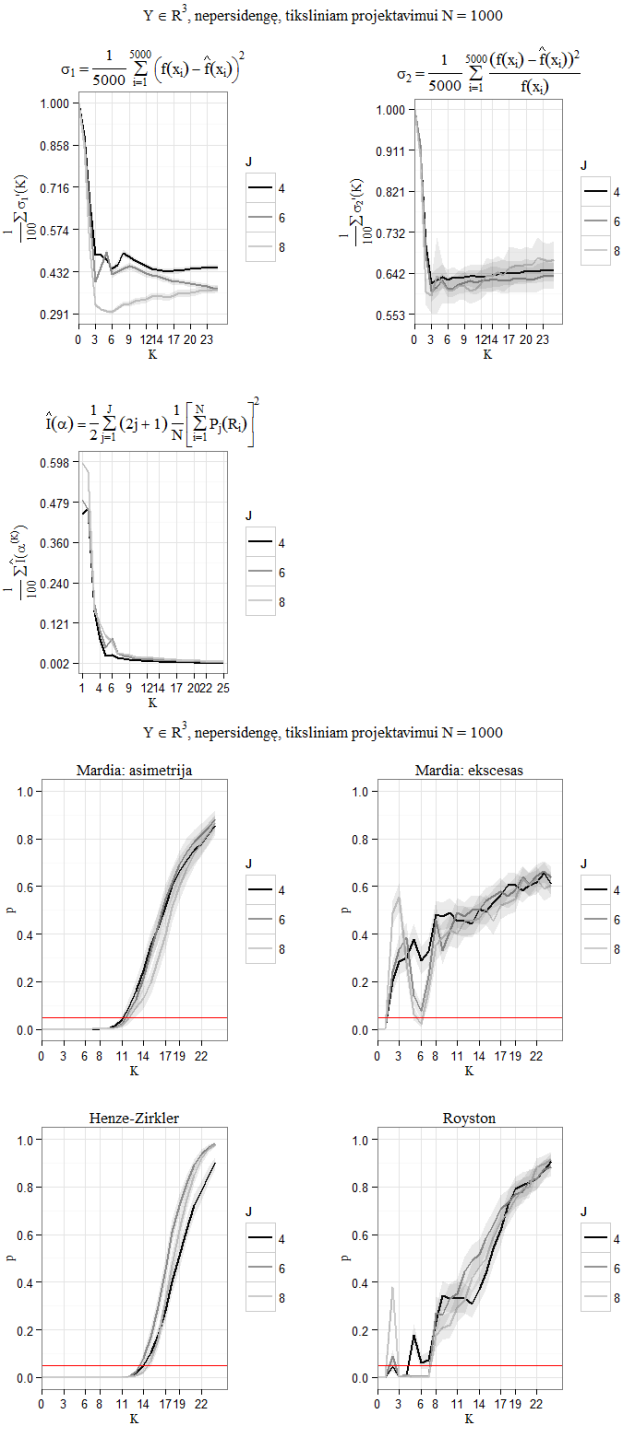
0.22 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 nepersidengusių klasių atveju



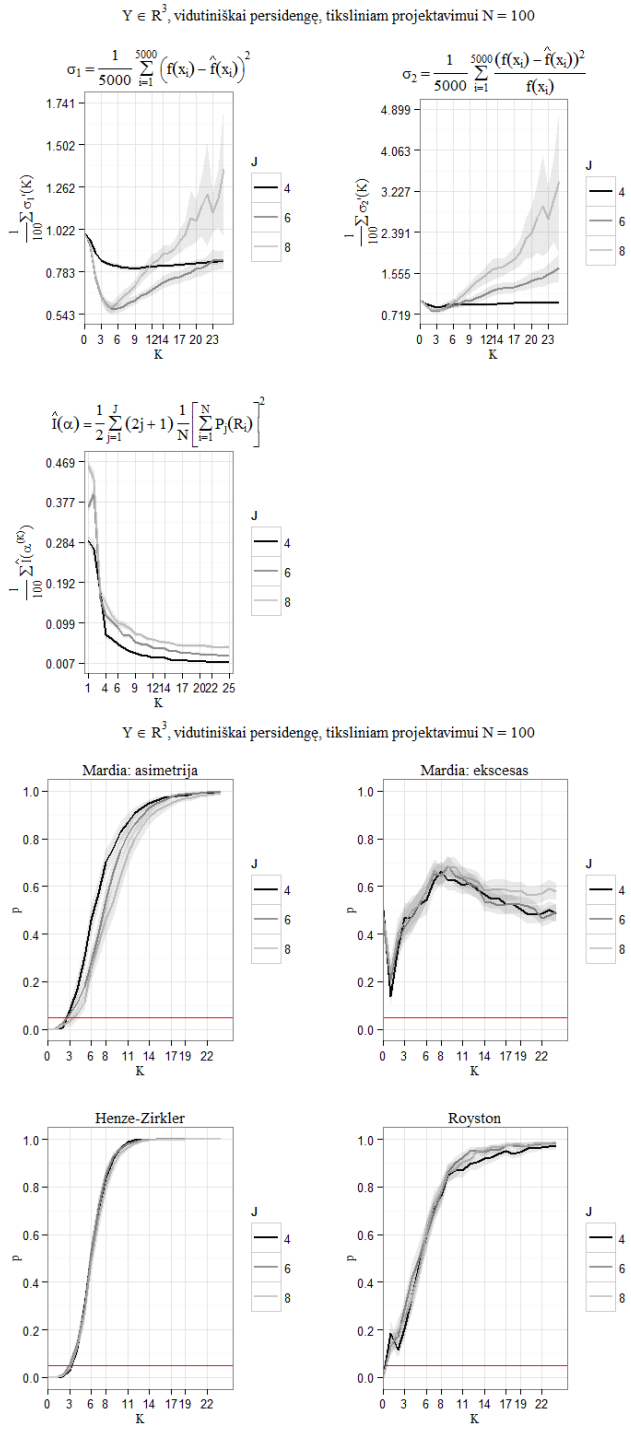
0.23 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 nepersidengusių klasių atveju



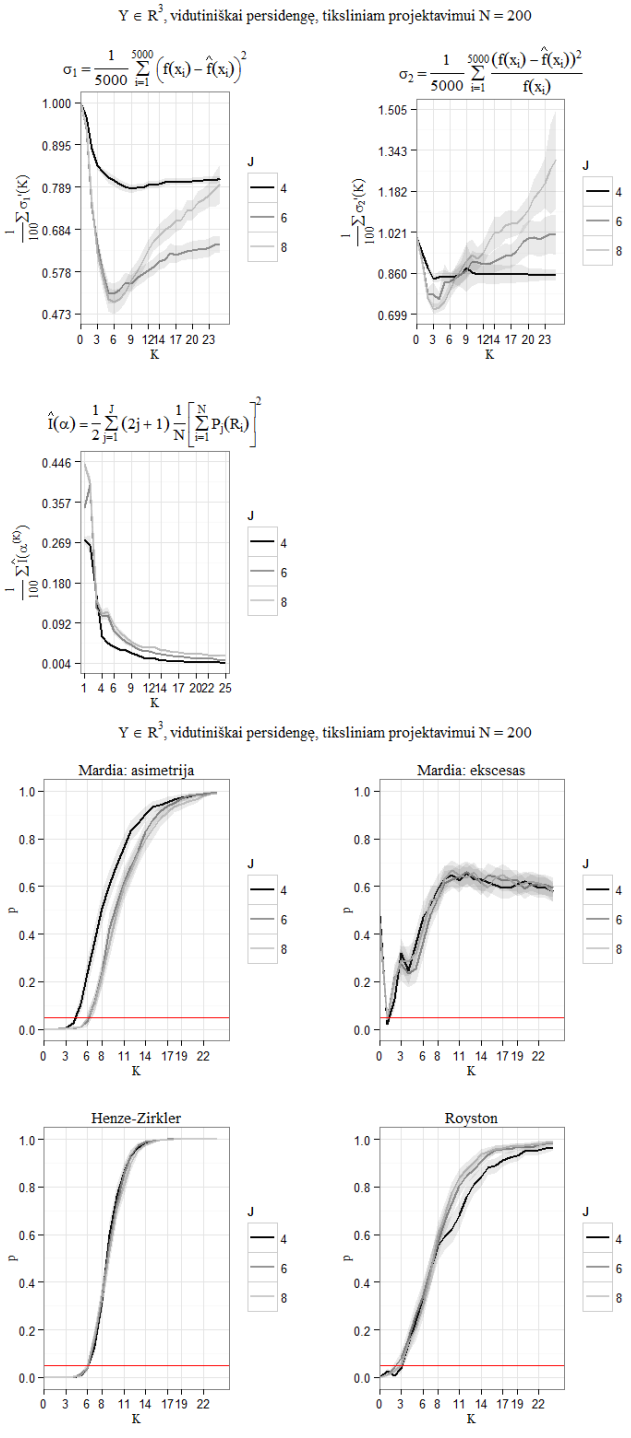
0.24 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 nepersidengusių klasių atveju



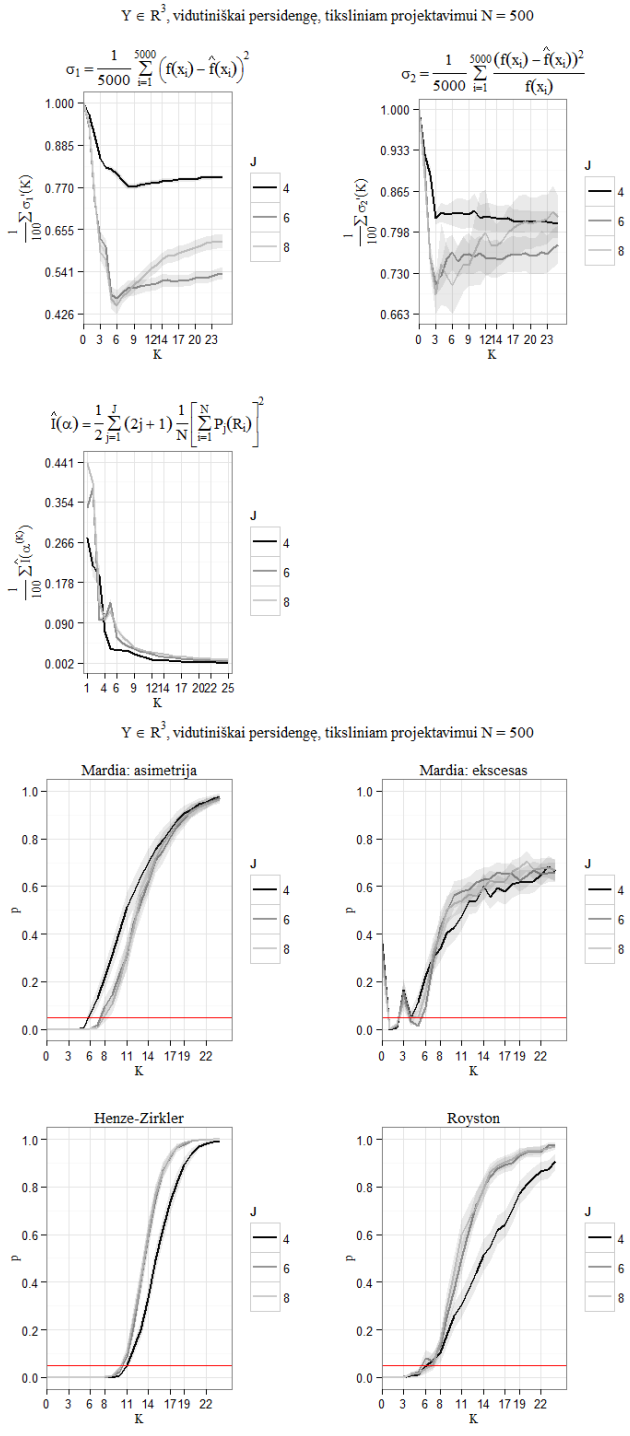
0.25 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 nepersidengusių klasių atveju



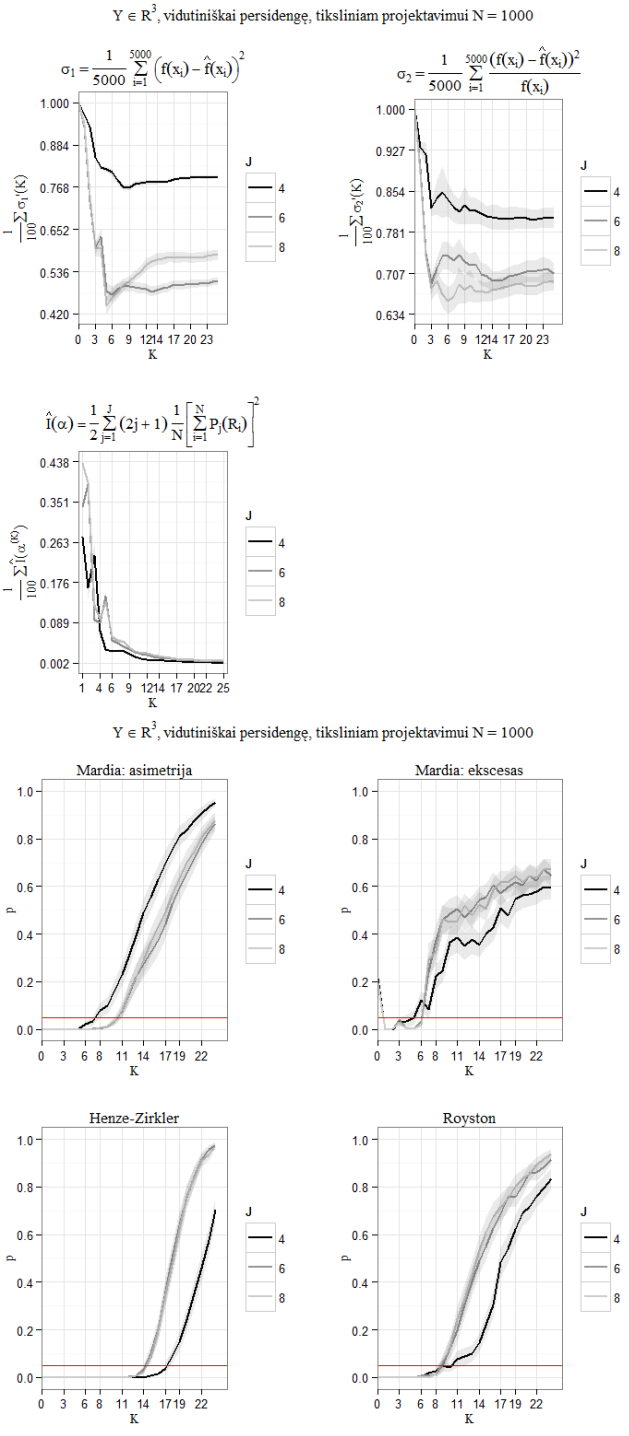
0.26 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 vidutiniškai persidengusių klasių atveju



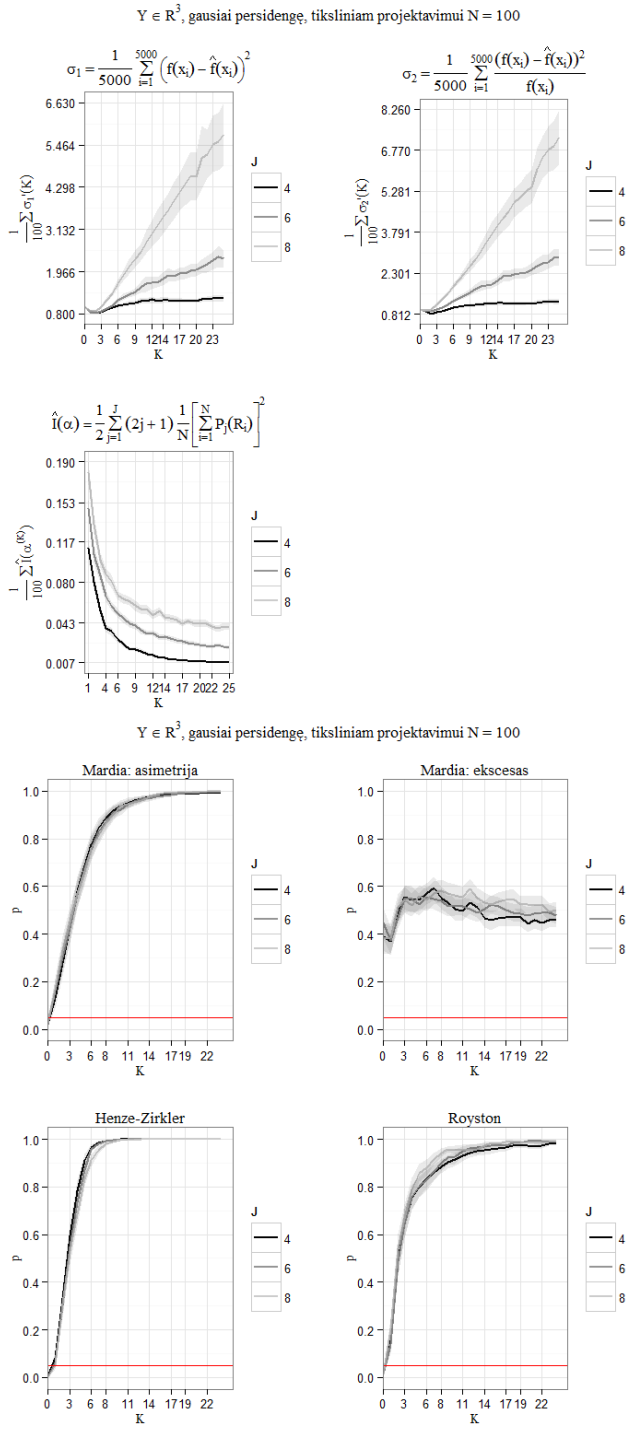
0.27 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 vidutiniškai persidengusių klasių atveju



0.28 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 vidutiniškai persidengusių klasių atveju

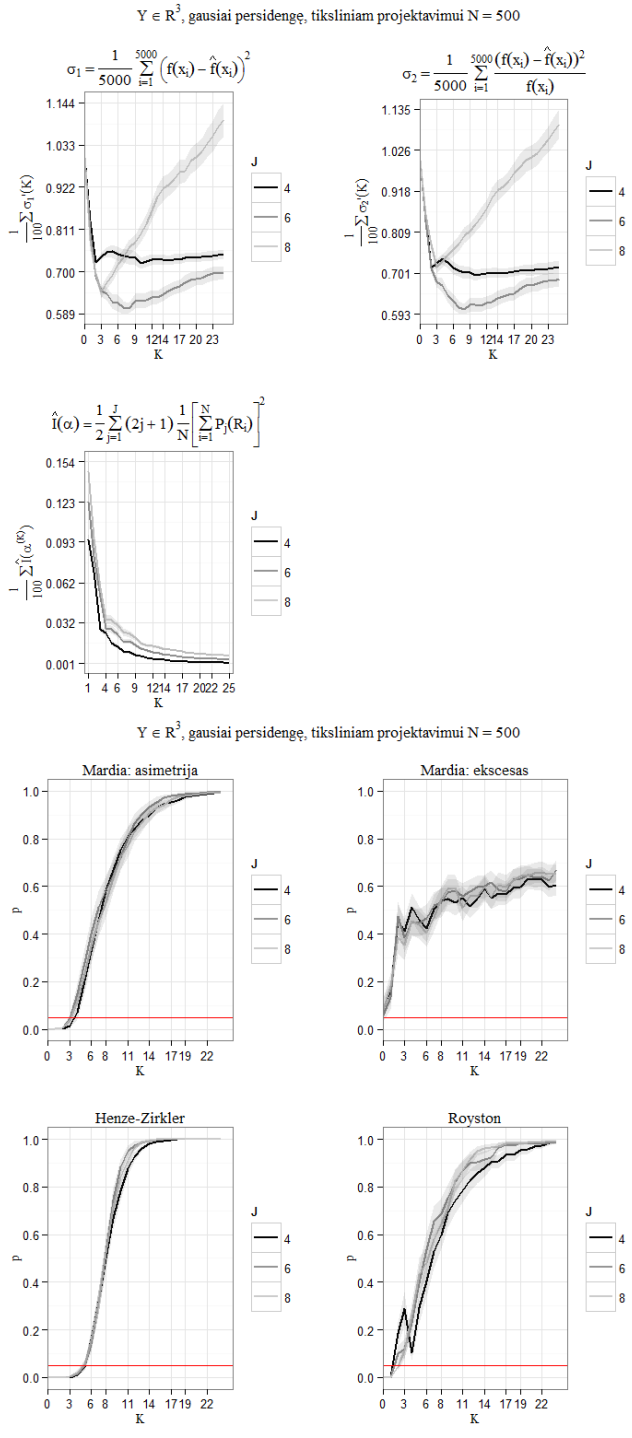


0.29 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 vidutiniškai persidengusių klasių atveju

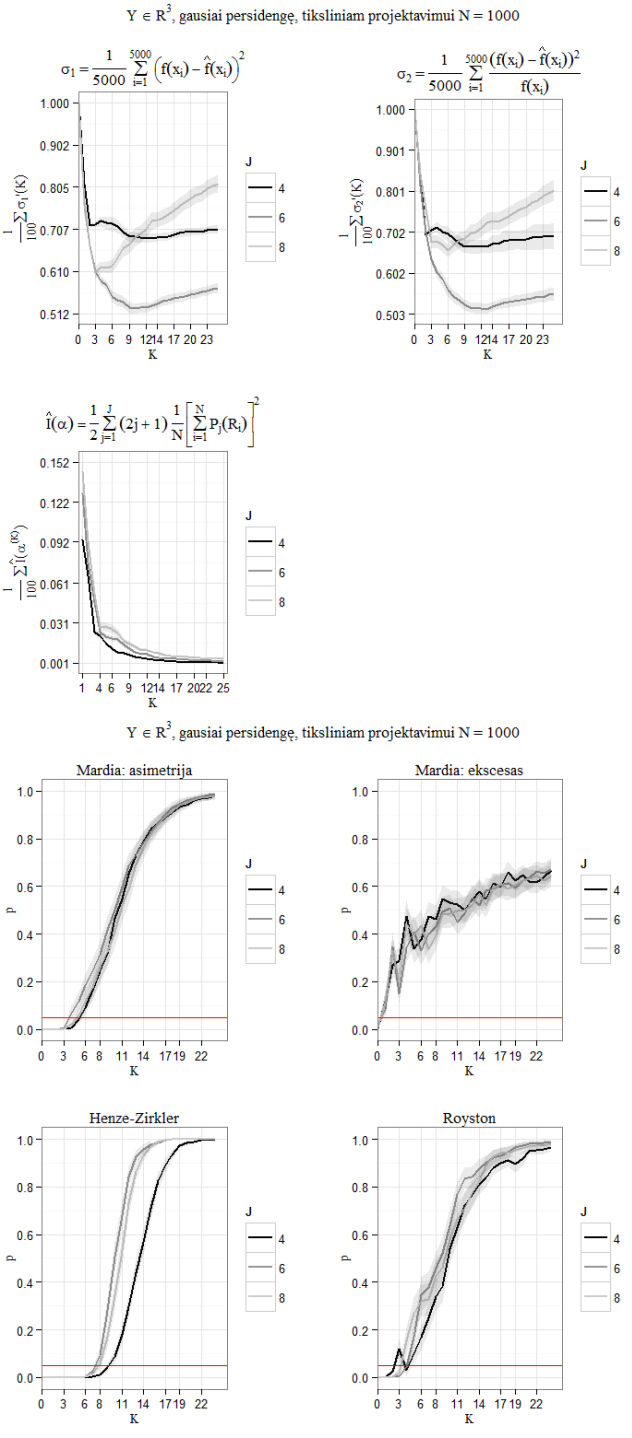


0.30 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 gausiai persidengusių klasių atveju

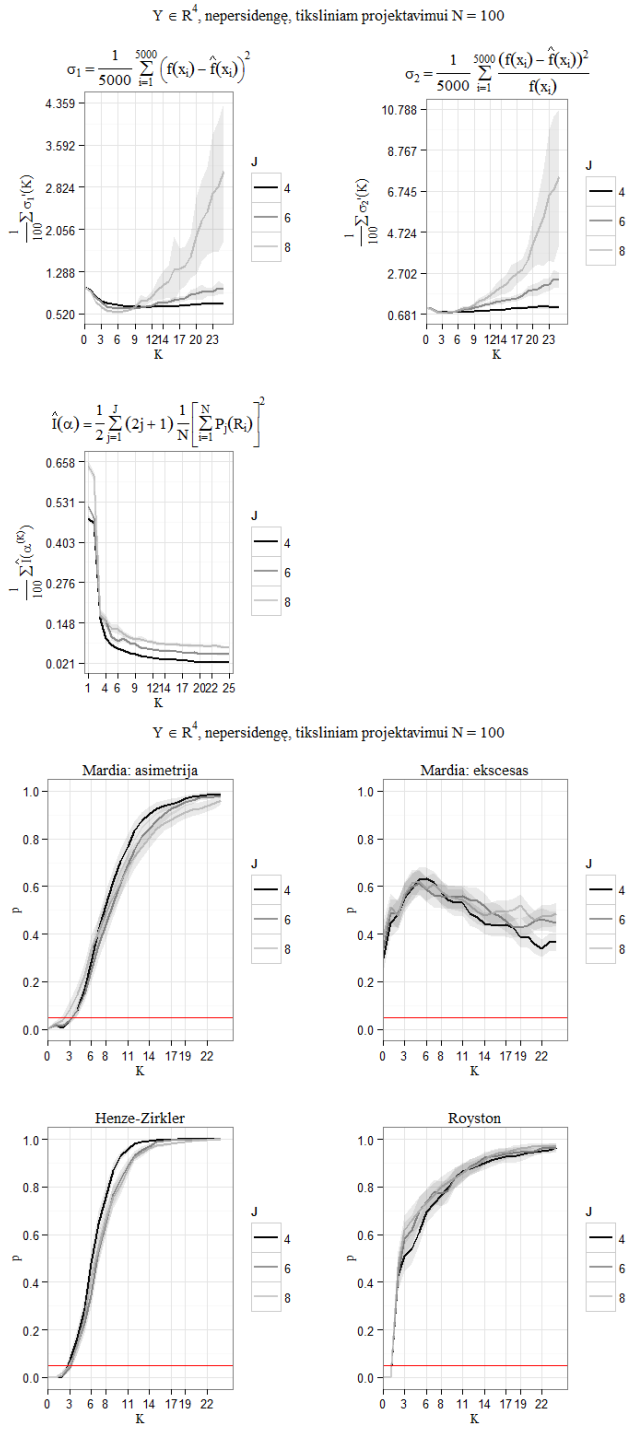
0.31 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 gausiai persidengusių klasių atveju



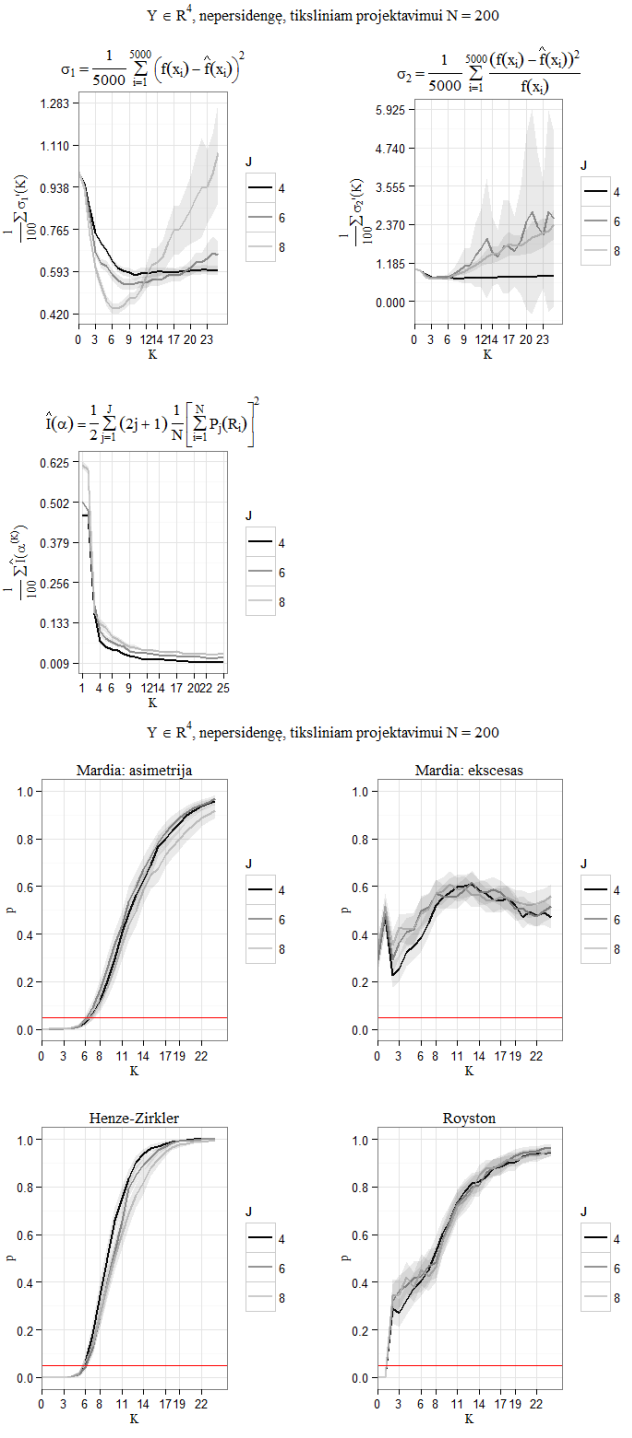
0.32 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 gausiai persidengusių klasių atveju



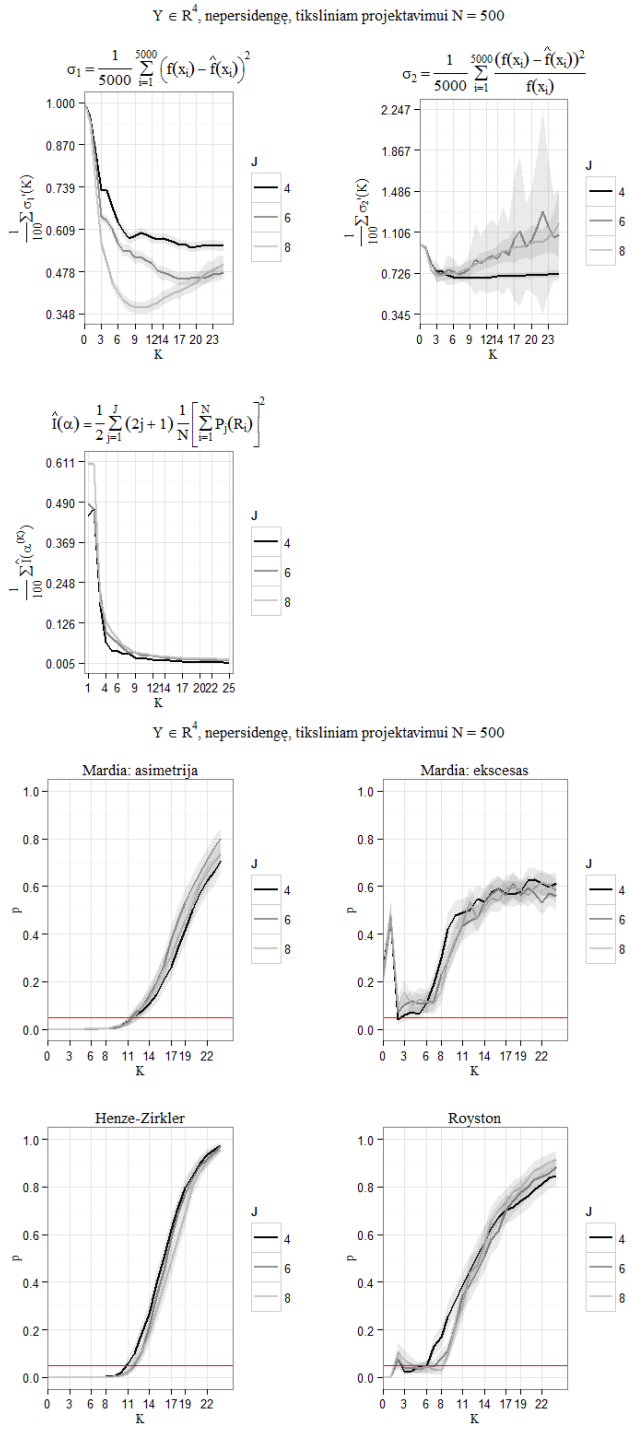
0.33 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^3 gausiai persidengusių klasių atveju



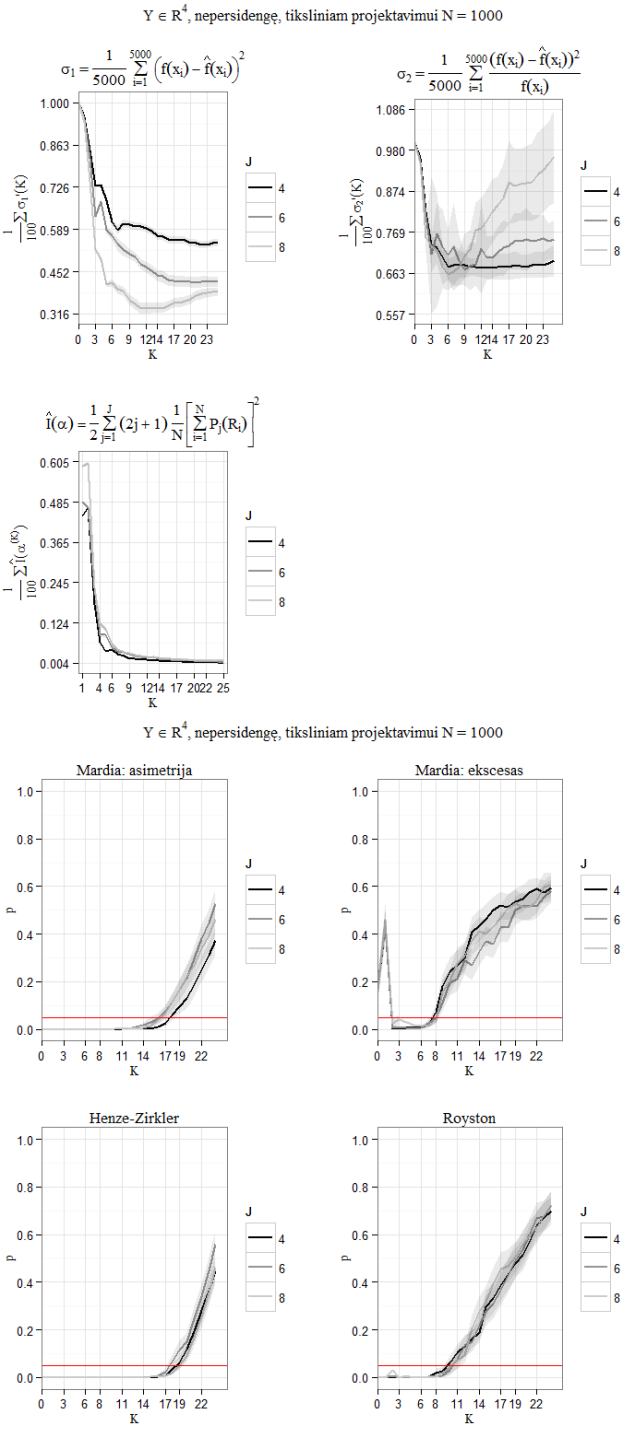
0.34 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 nepersidengusių klasių atveju



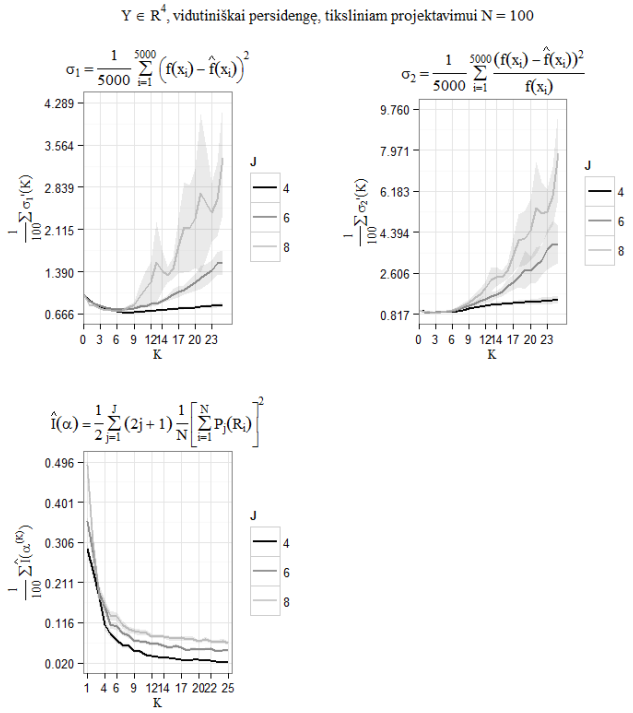
0.35 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 nepersidengusių klasių atveju

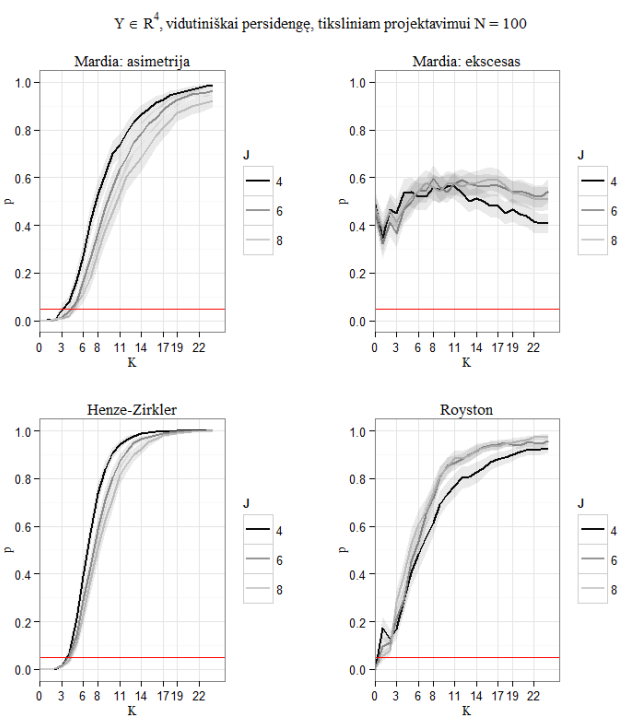


0.36 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 nepersidengusių klasių atveju

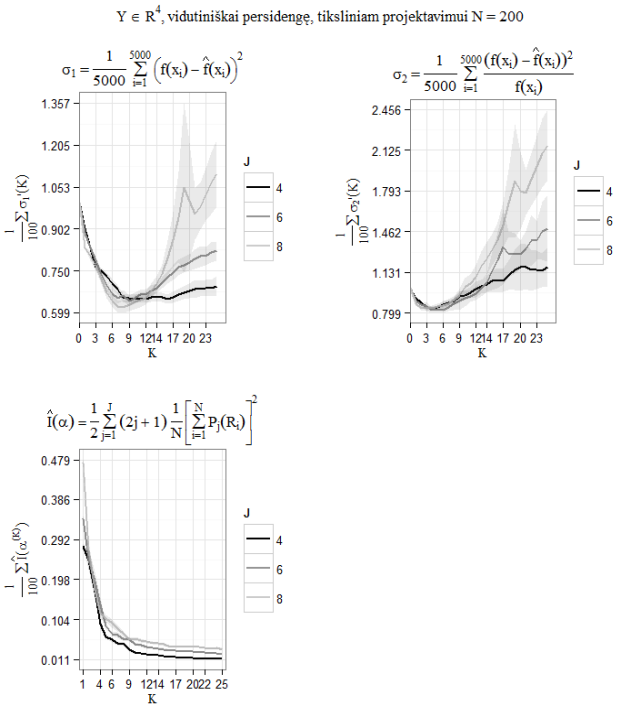


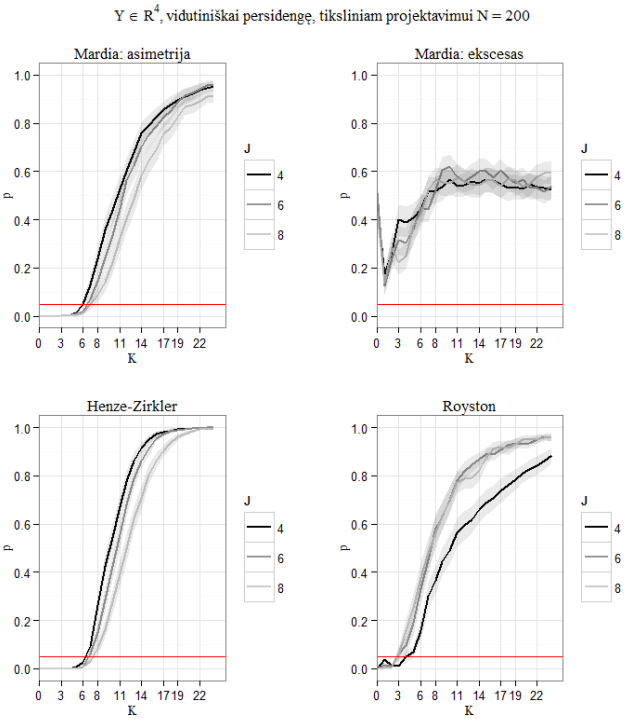
0.37 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 nepersidengusių klasių atveju



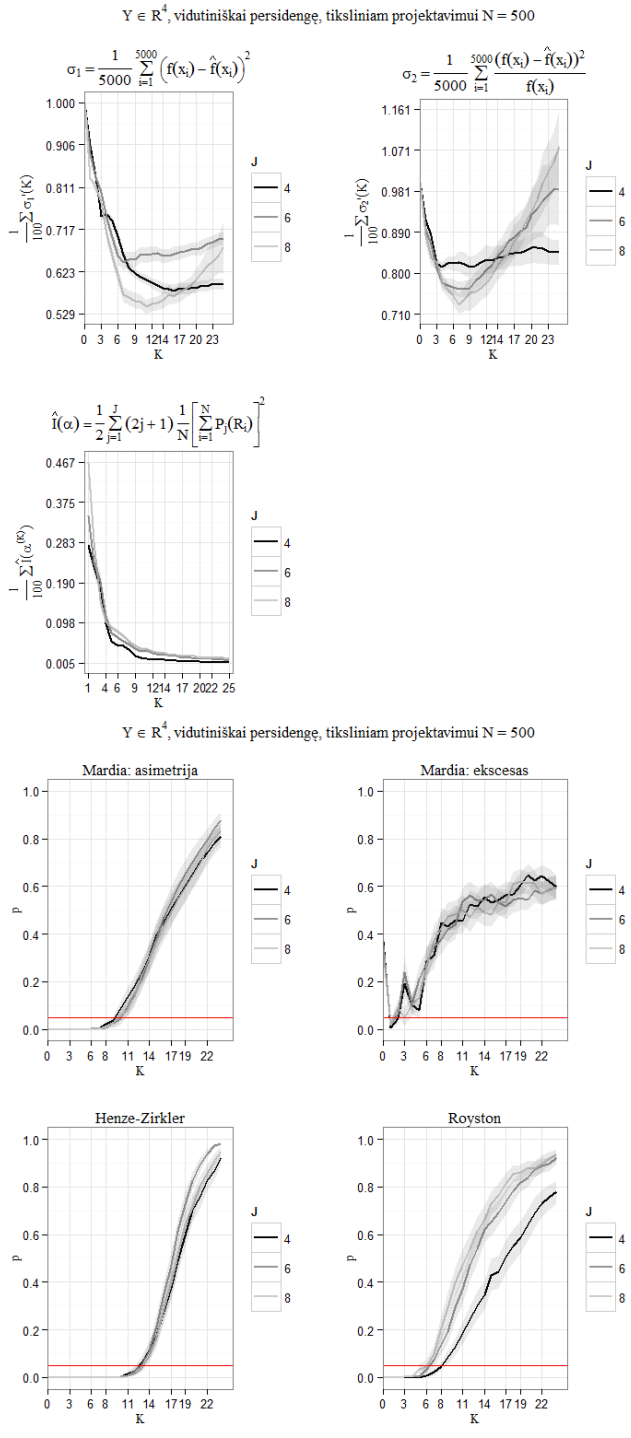


0.38 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 vidutiniškai persidengusių klasių atveju

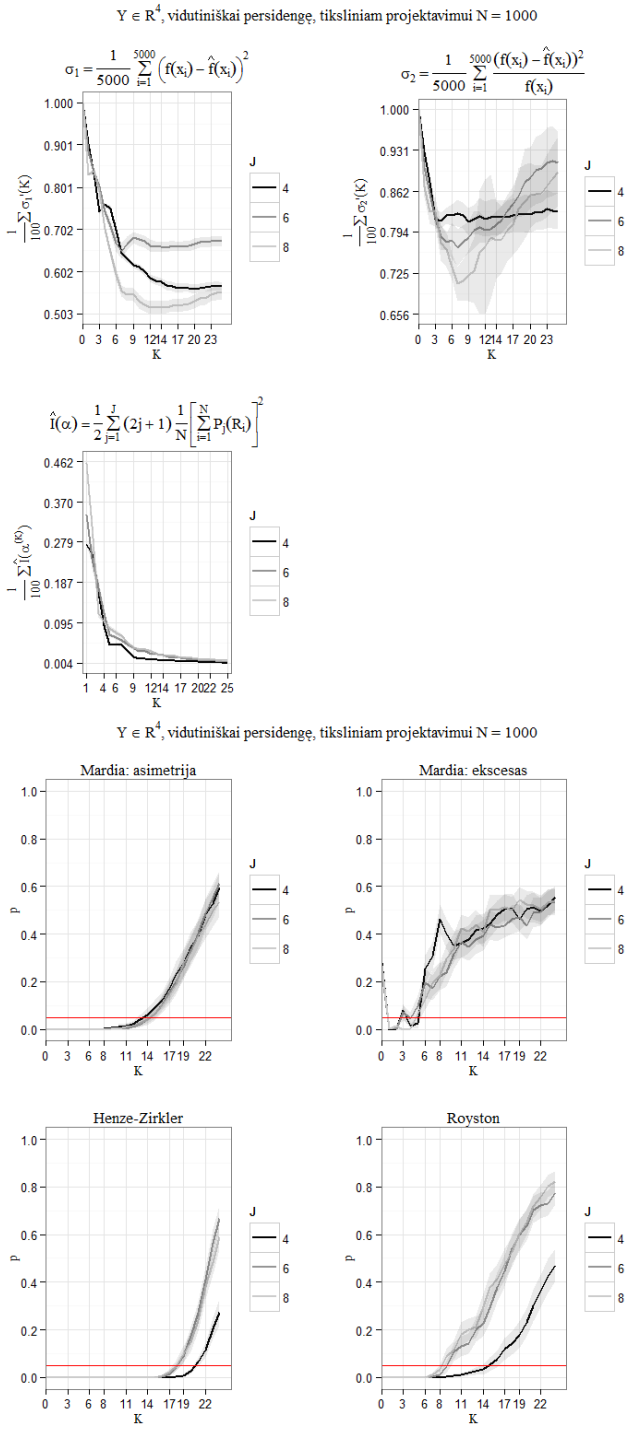




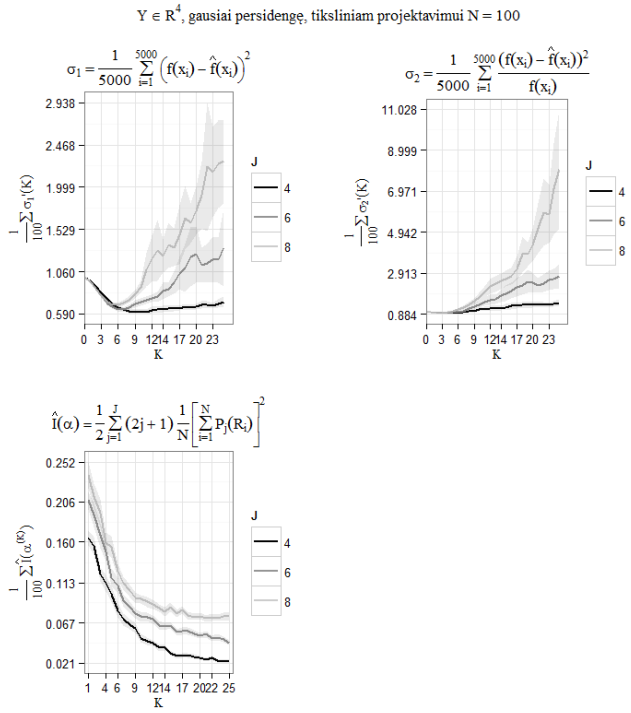
0.39 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 vidutiniškai persidengusių klasių atveju

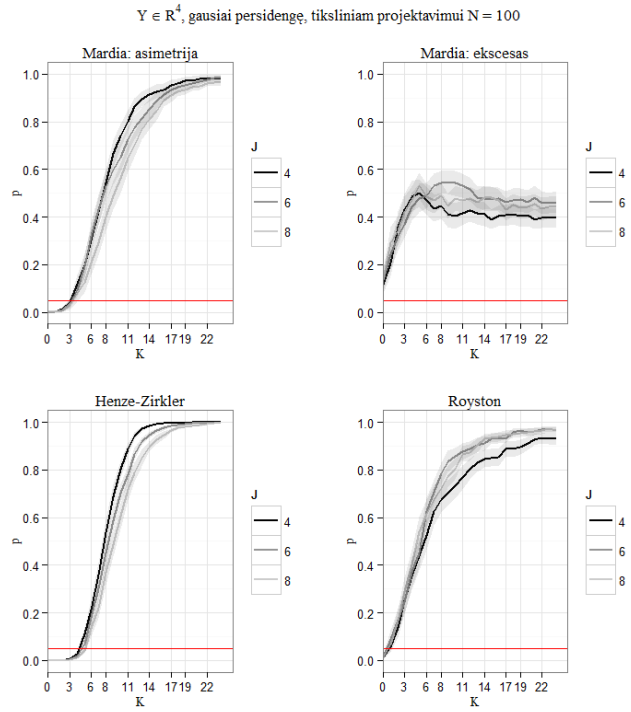


0.40 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 vidutiniškai persidengusių klasių atveju

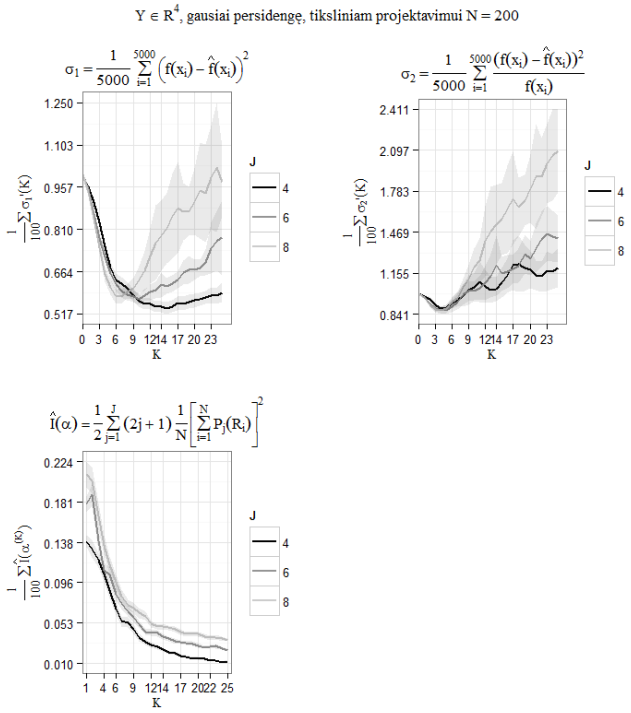


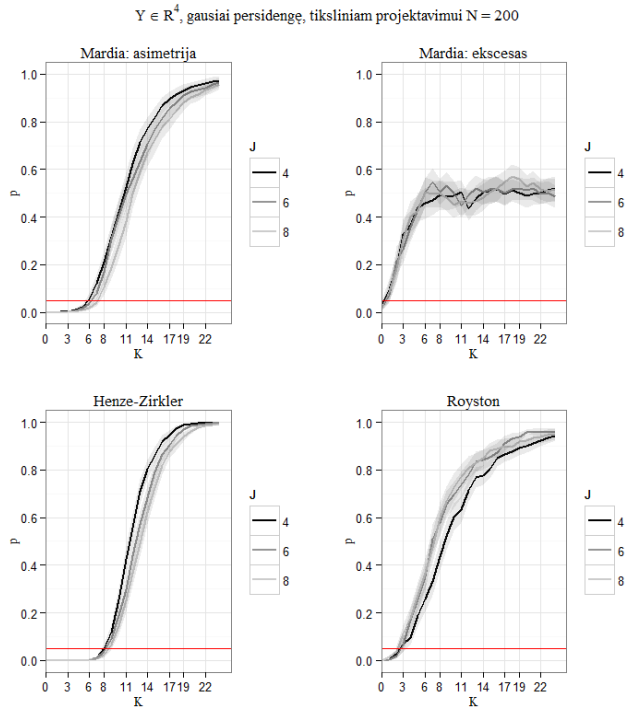
0.41 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 vudišiniškai persidengusių klasių atveju



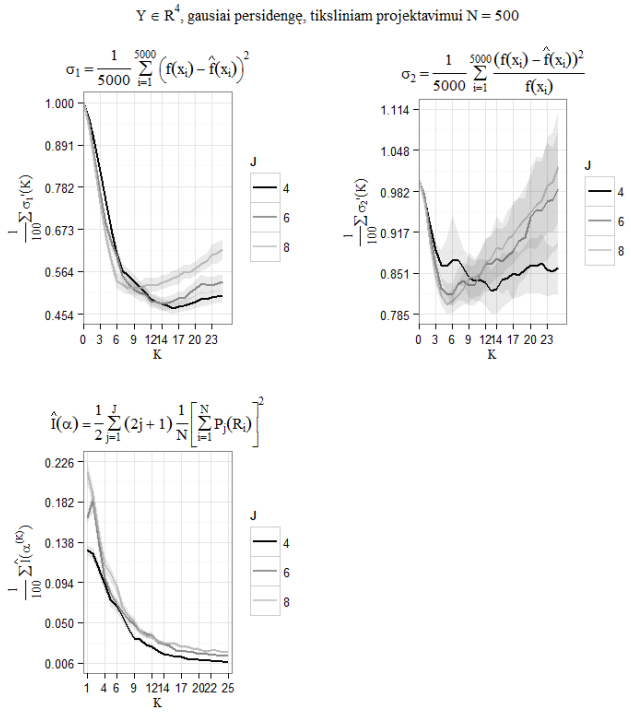


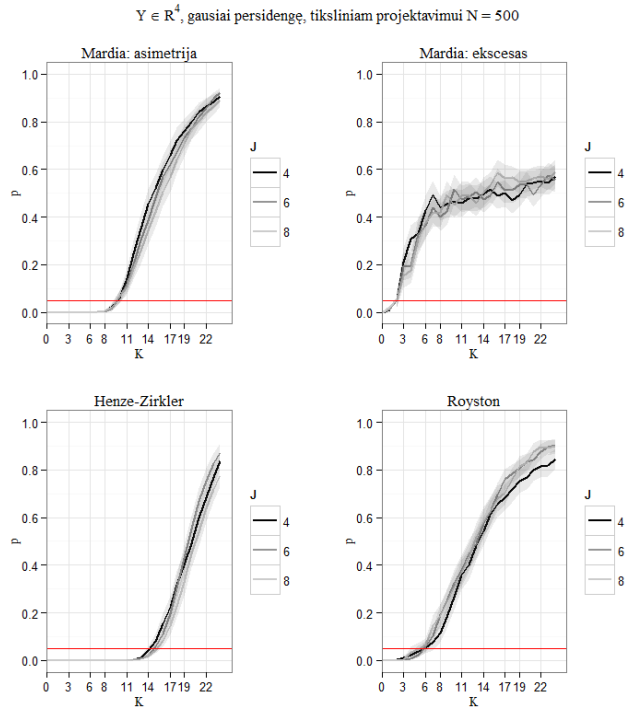
0.42 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 gausiai persidengusių klasių atveju



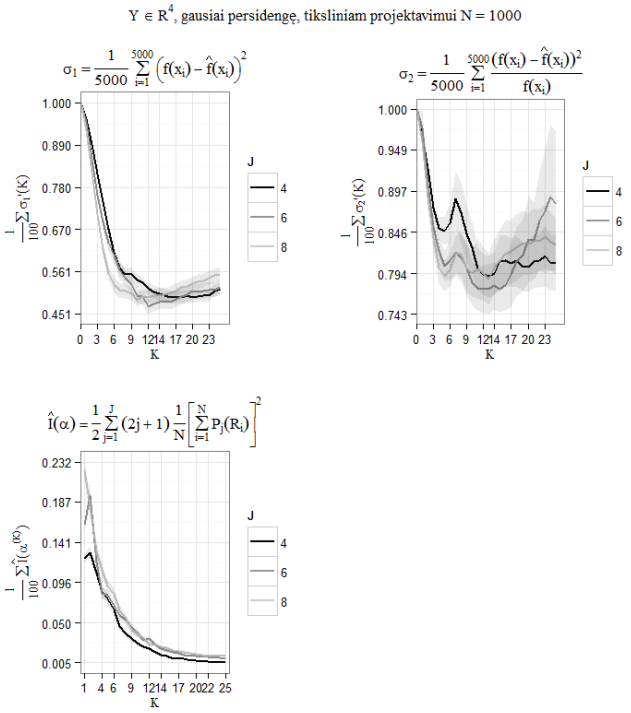


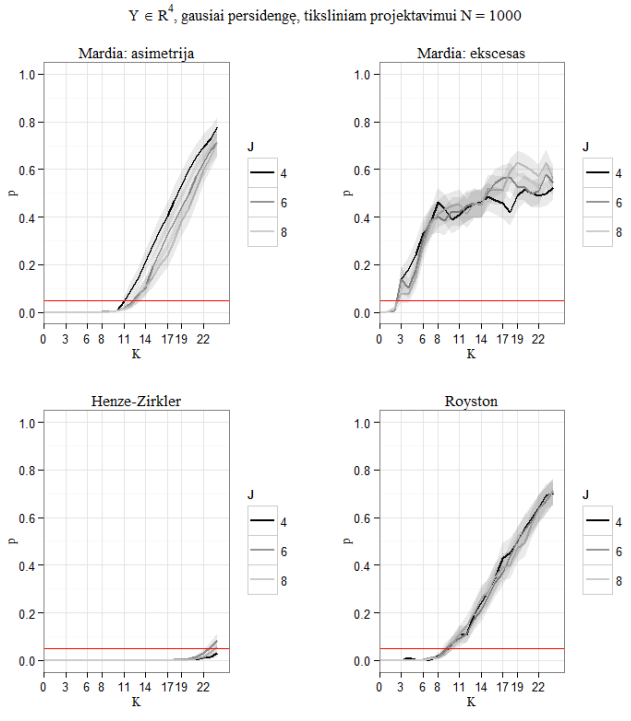
0.43 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 gausiai persidengusių klasių atveju



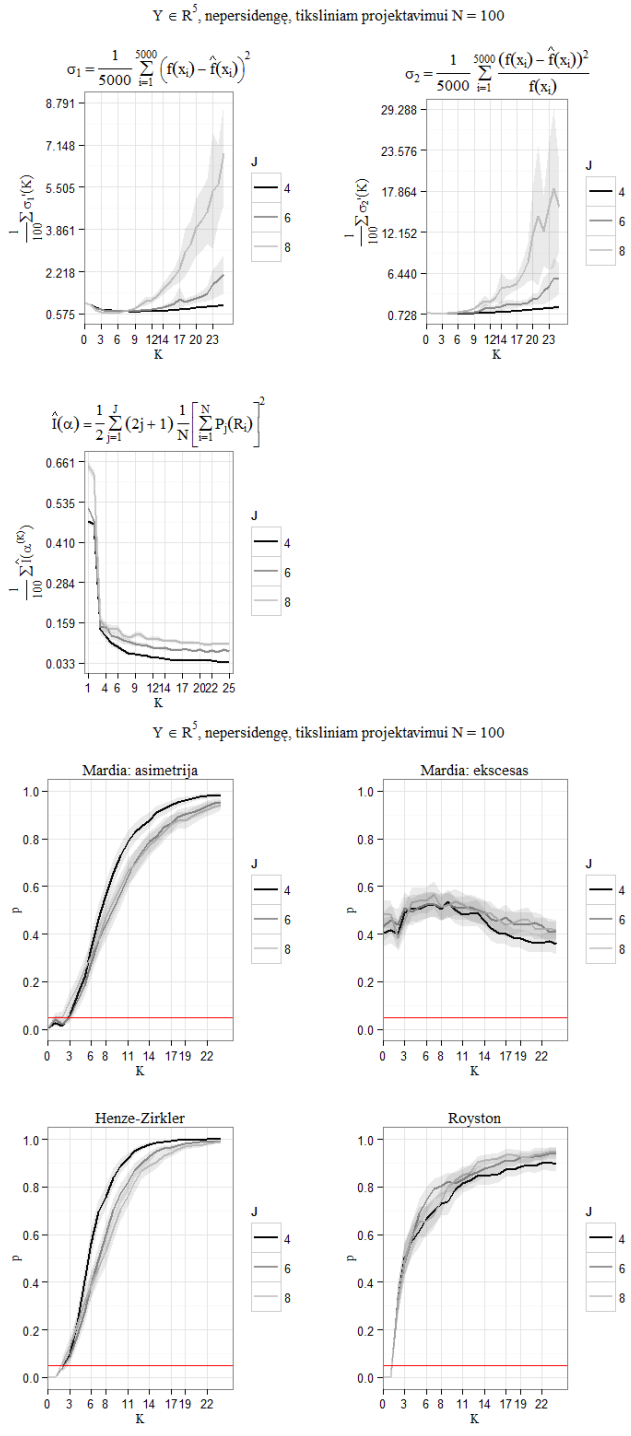


0.44 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 gausiai persidengusių klasių atveju

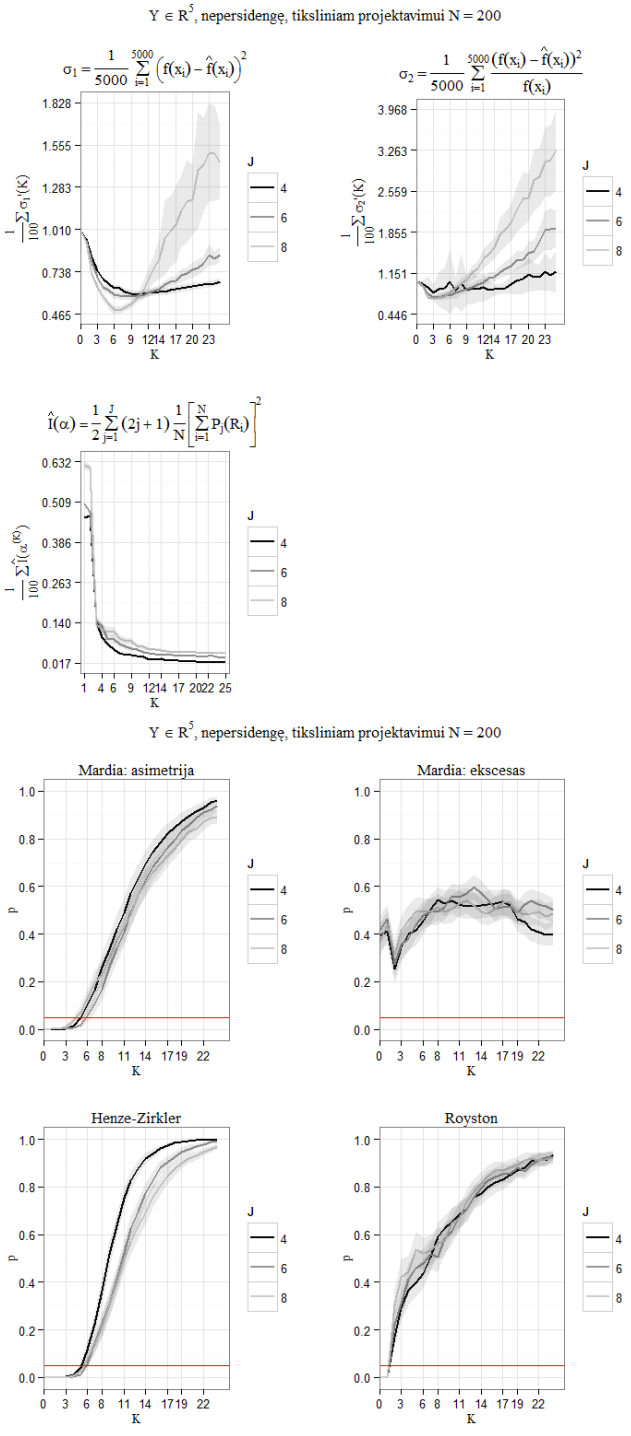




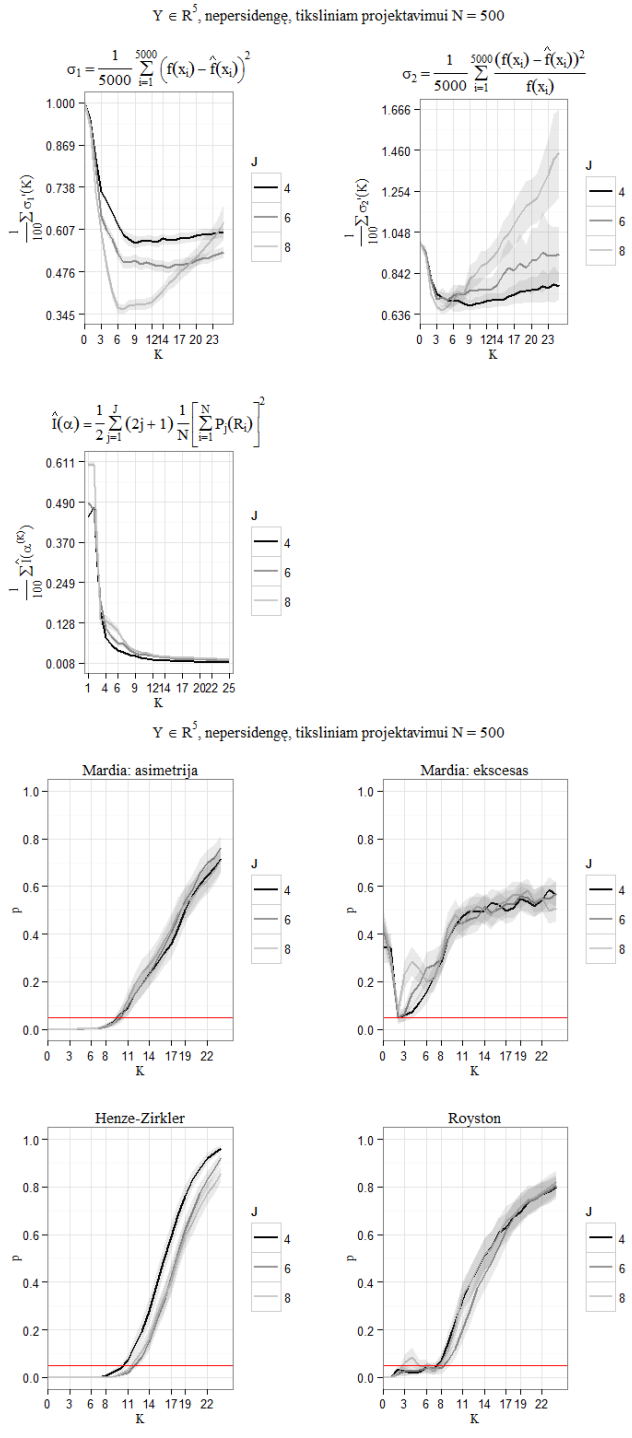
0.45 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^4 gausiai persidengusių klasių atveju



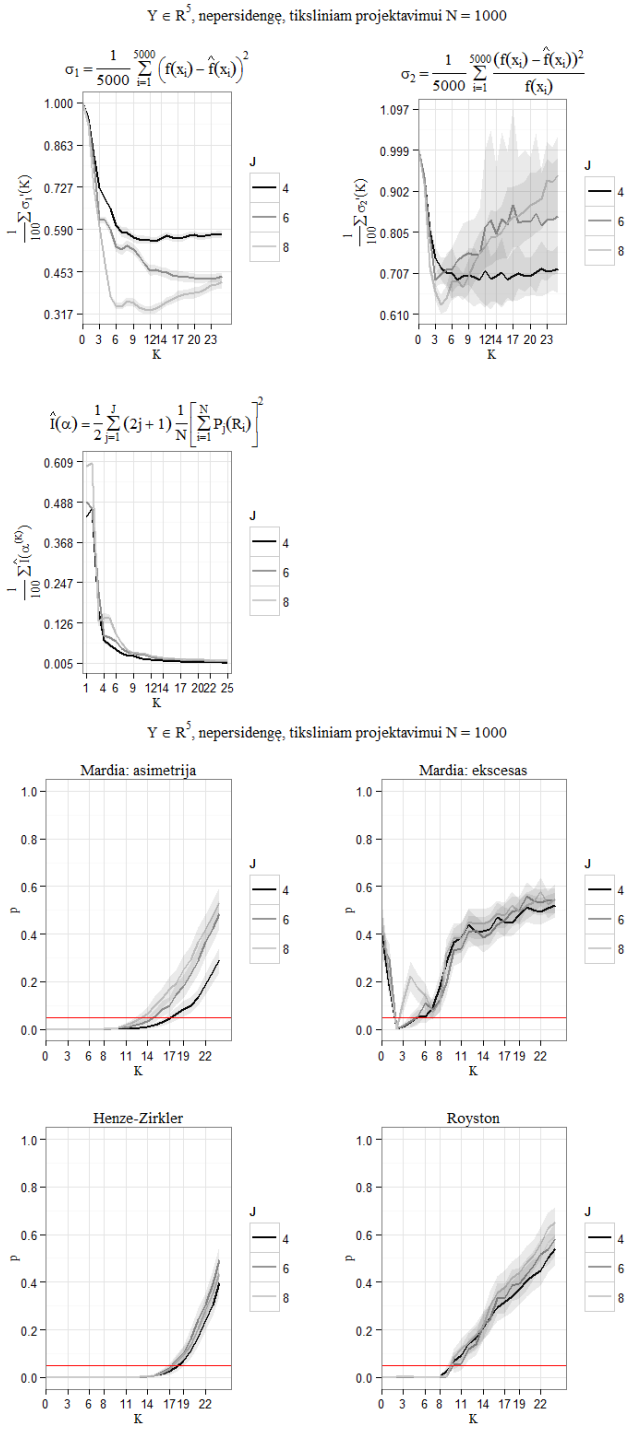
0.46 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 nepersidenguių klasių atveju



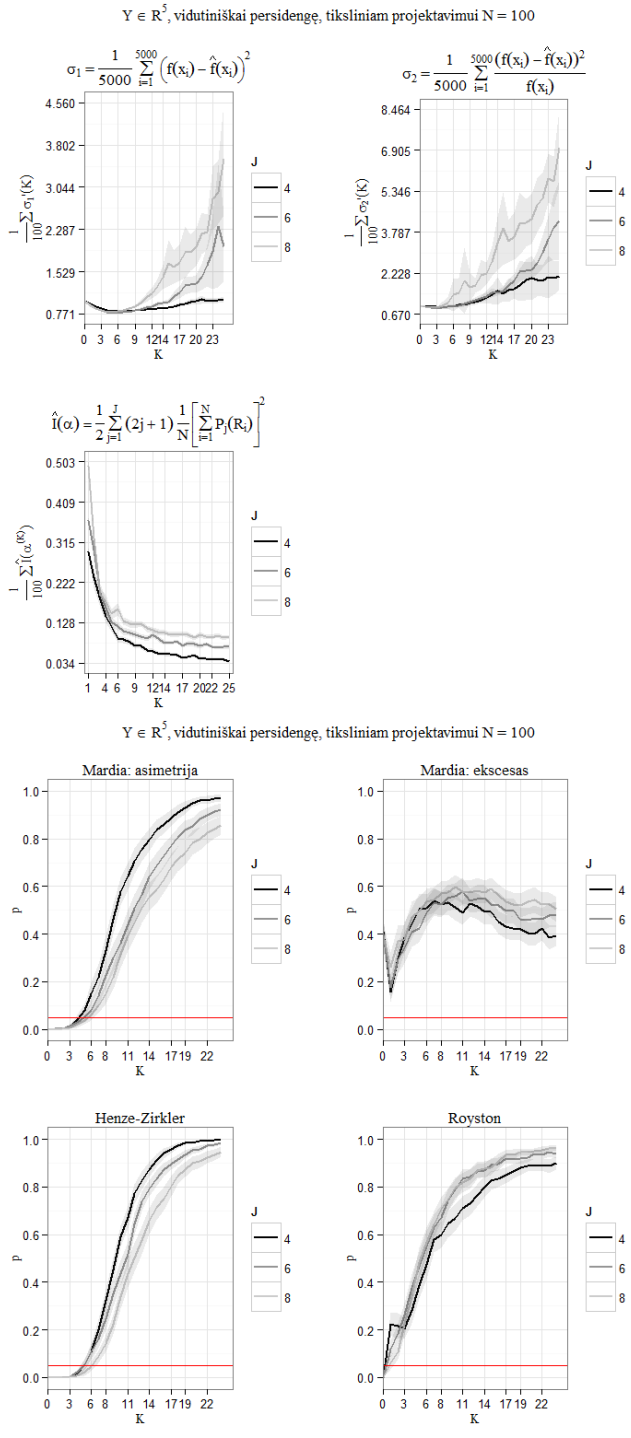
0.47 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 nepersidenguių klasių atveju



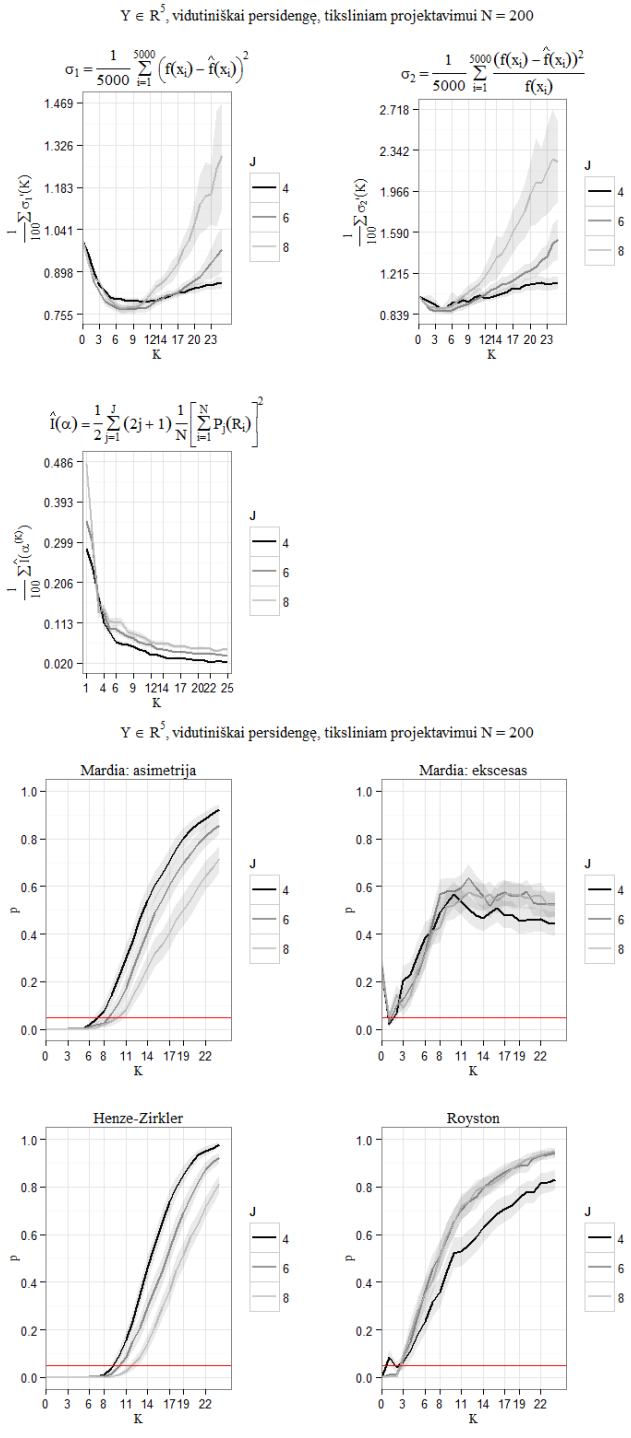
0.48 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 nepersidengusių klasių atveju



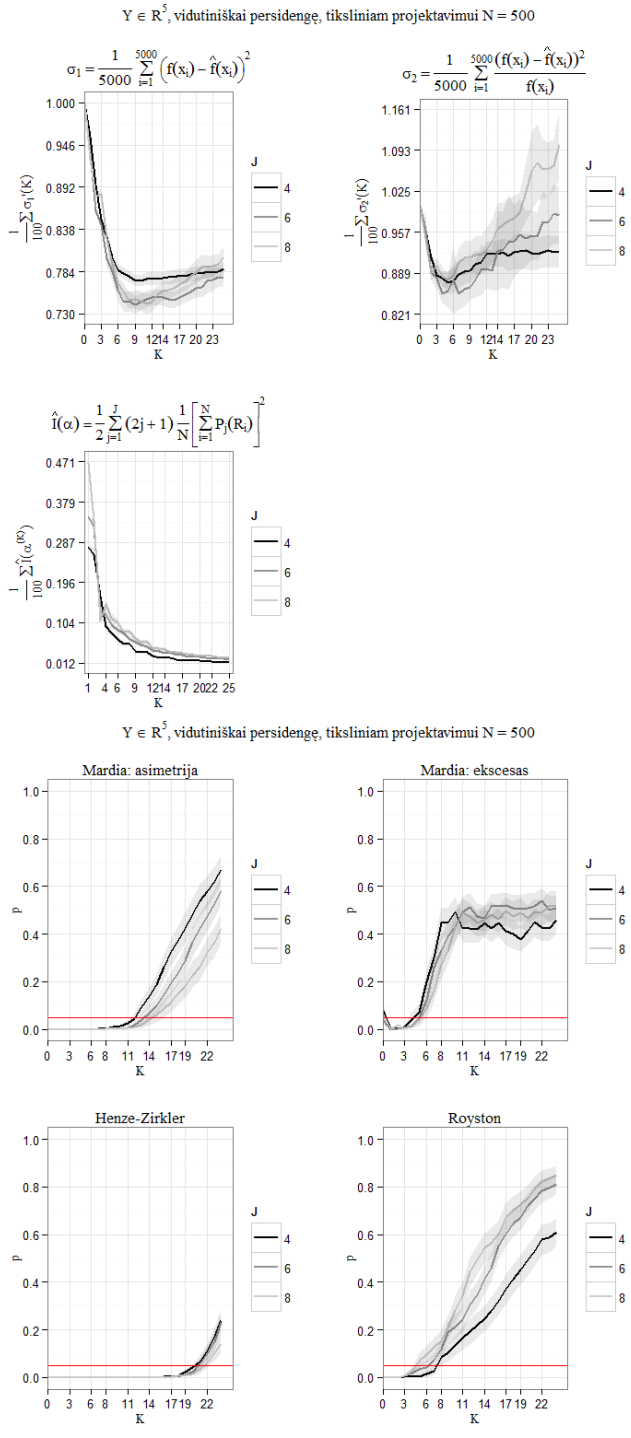
0.49 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 nepersidengusių klasių atveju



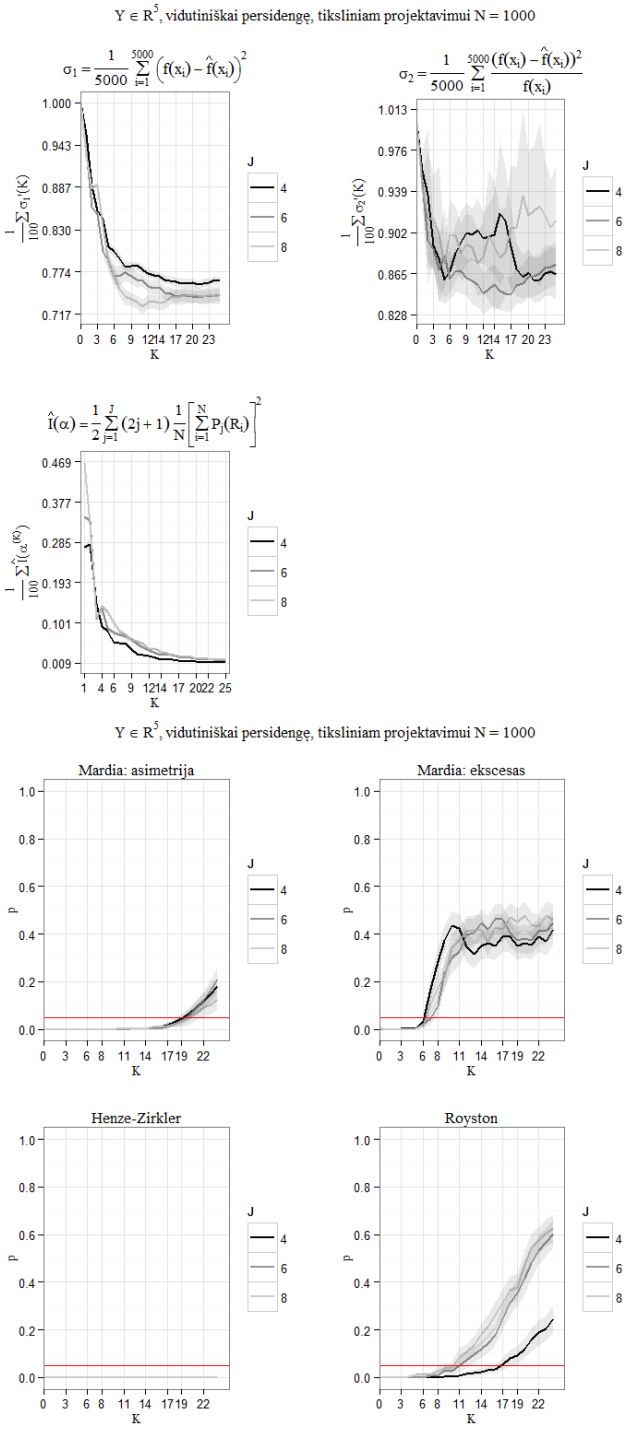
0.50 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 vidutiniškai persidengusių klasių atveju



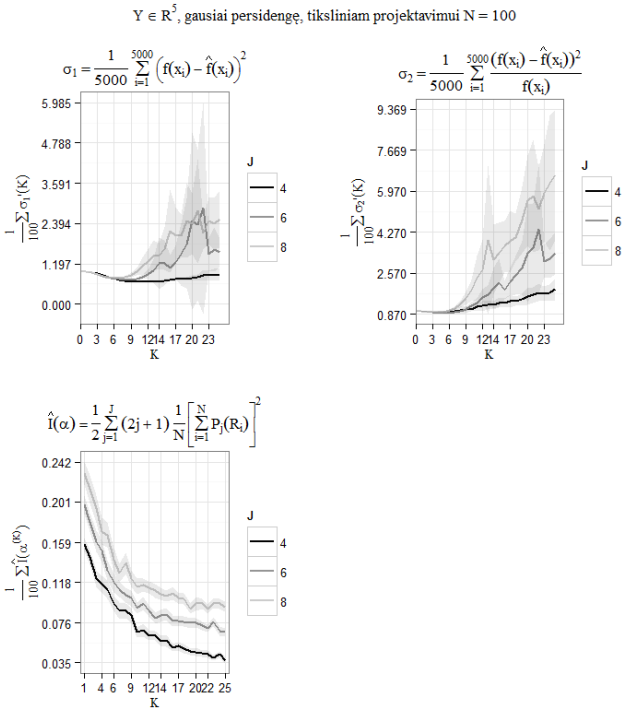
0.51 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 vidutiniškai persidengusių klasių atveju

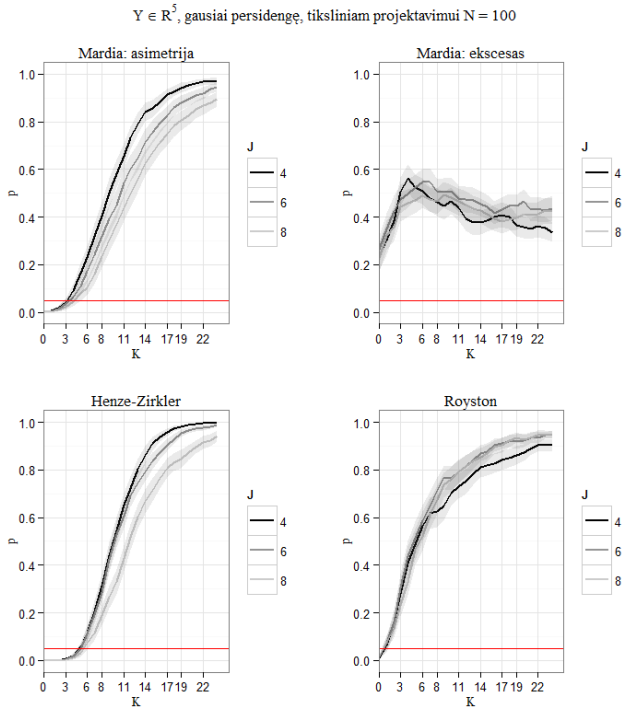


0.52 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 vidutiniškai persidengusių klasių atveju

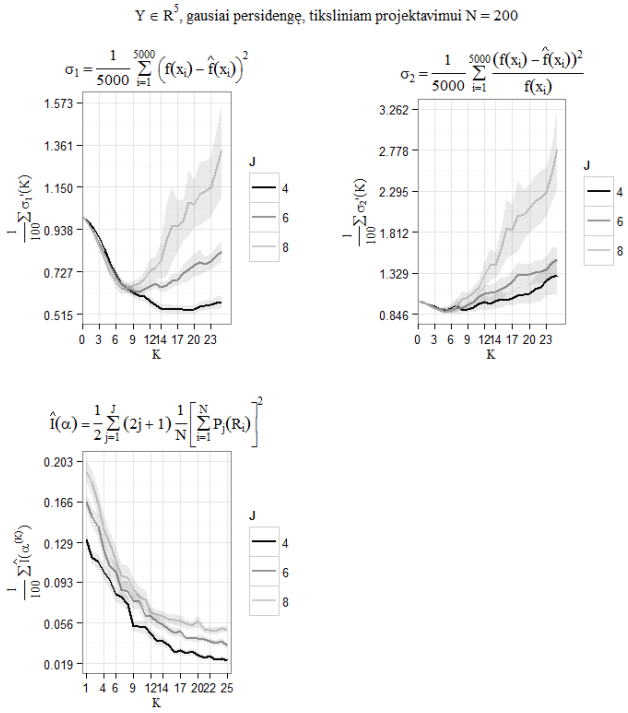


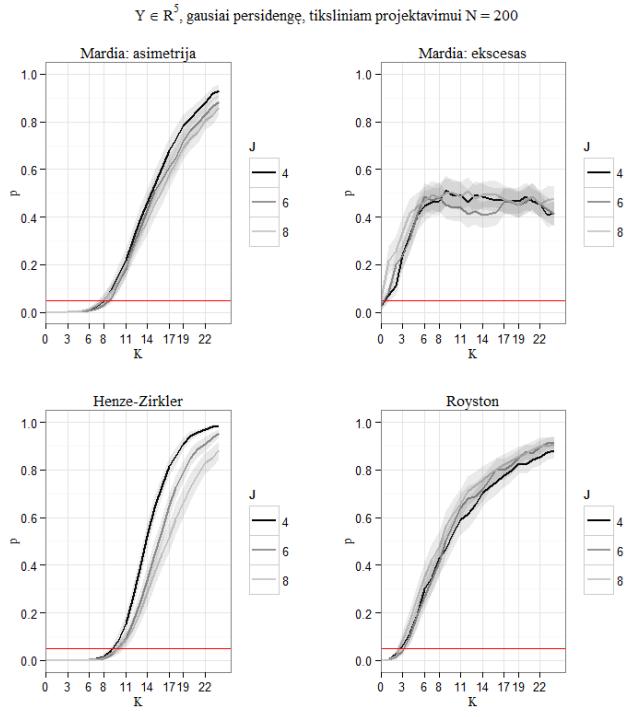
0.53 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 vidutiniškai persidengusių klasių atveju



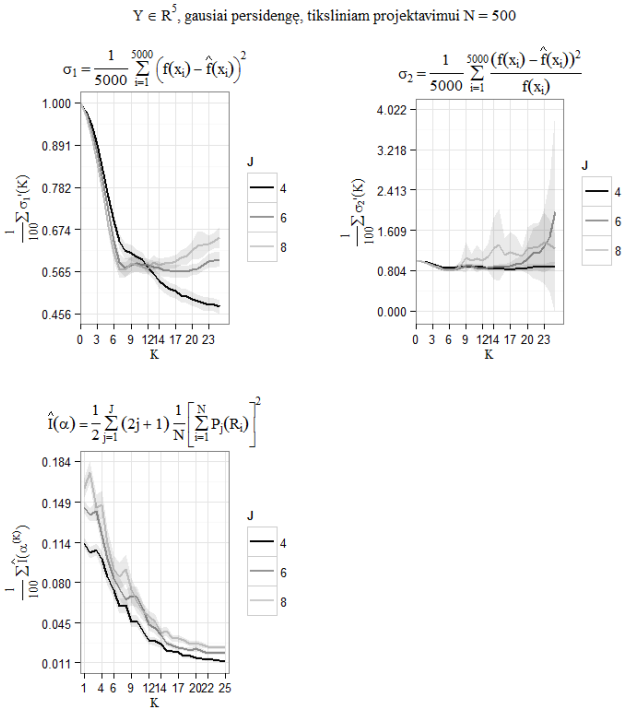


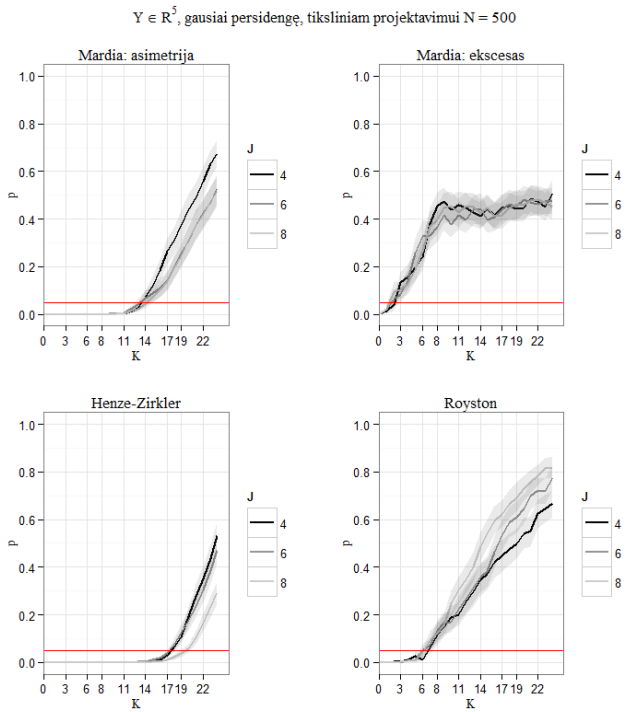
0.54 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 gausiai persidengusių klasių atveju



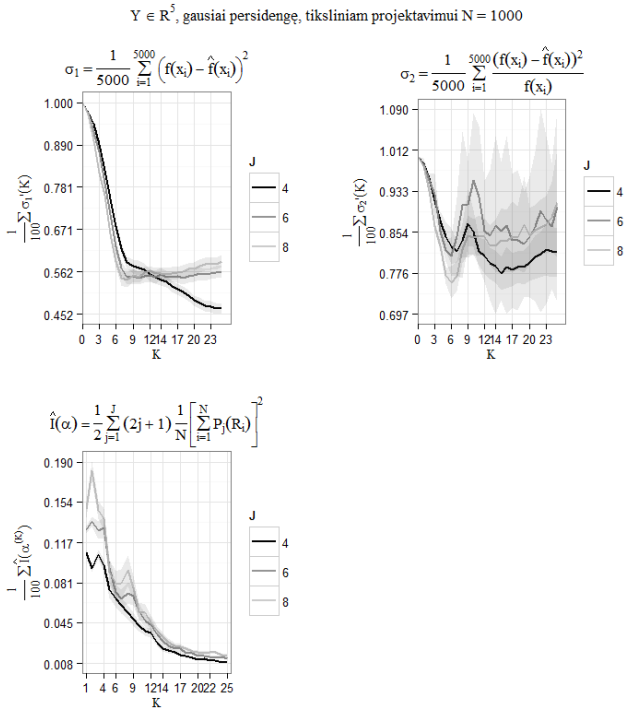


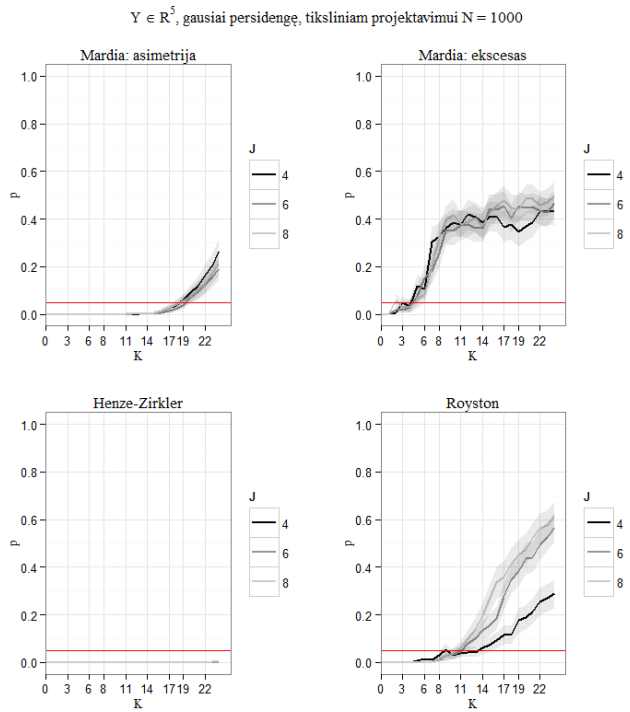
0.55 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 gausiai persidengusių klasių atveju





0.56 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 gausiai persidengusių klasių atveju

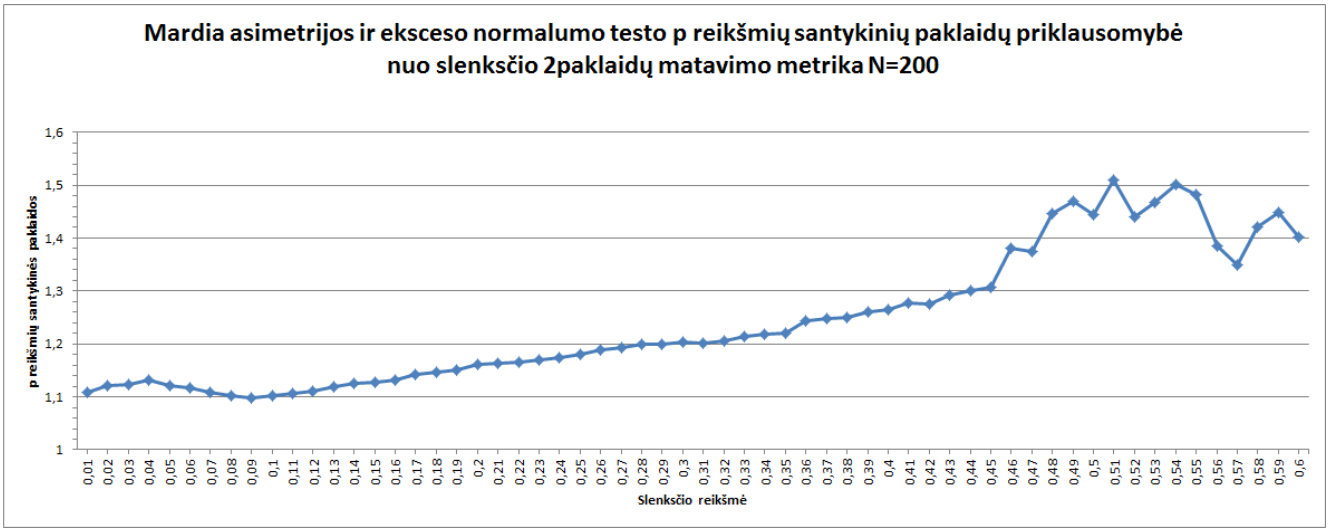




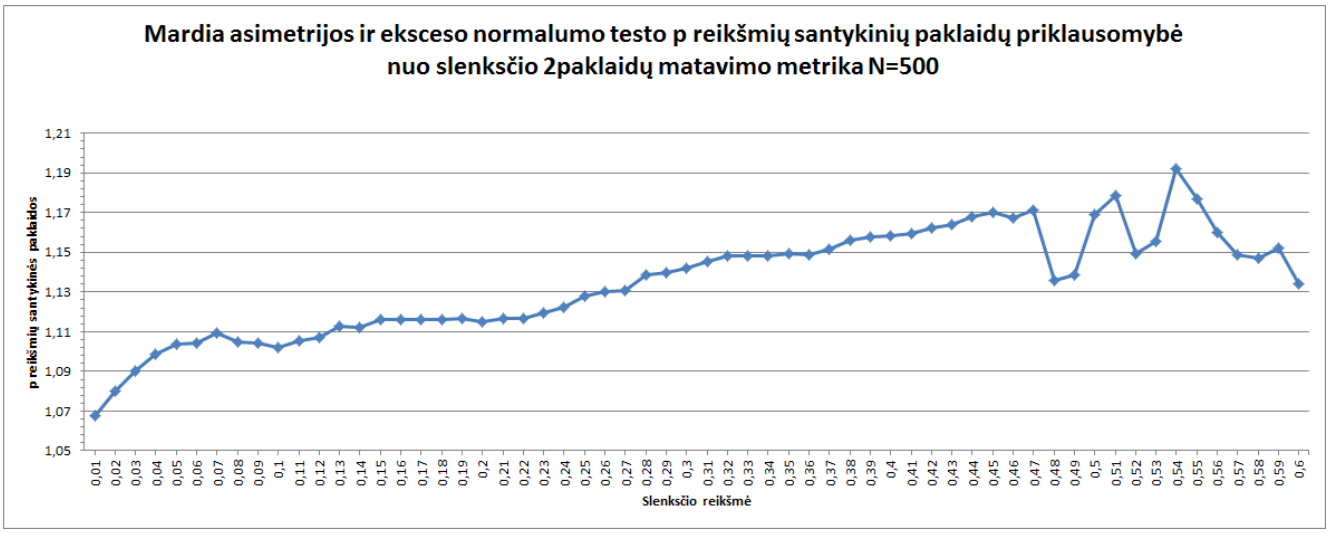
0.57 pav. Projektavimo indekso, paklaidų ir normalumo testų p reikšmių grafikai R^5 gausiai persidengusių klasių atveju

4 Priedas

SANTYKINIŲ PAKLAIDŲ, PRIKLAUSOMYBĖS NUO SLENKSČIO, GRAFIKAI

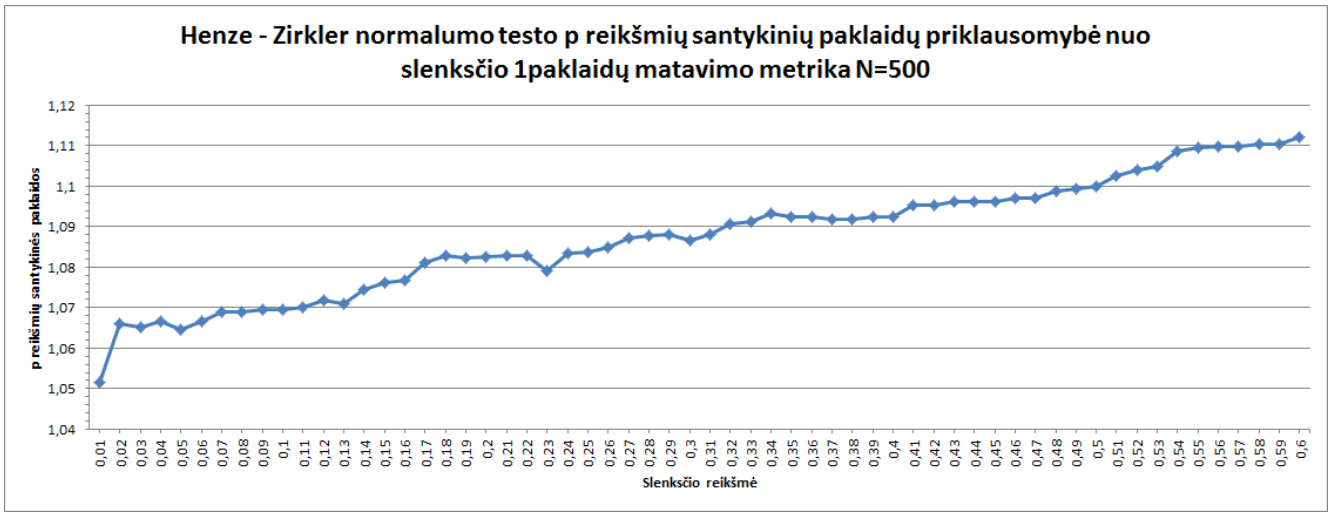


0.58 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 200$

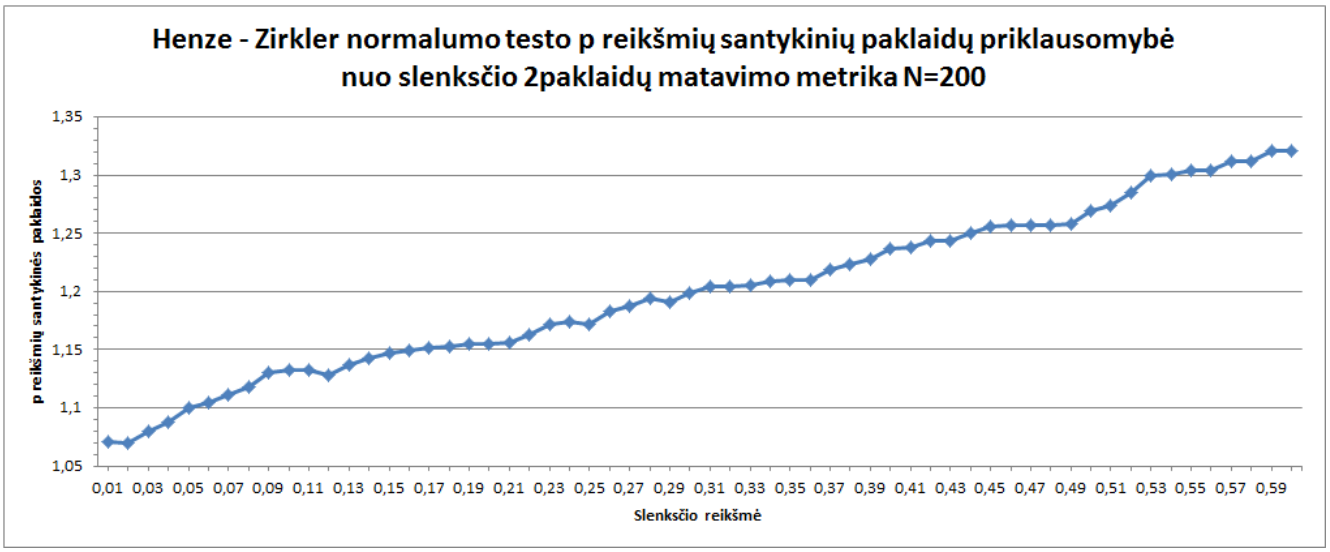


0.59 pav. Mardia normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 500$

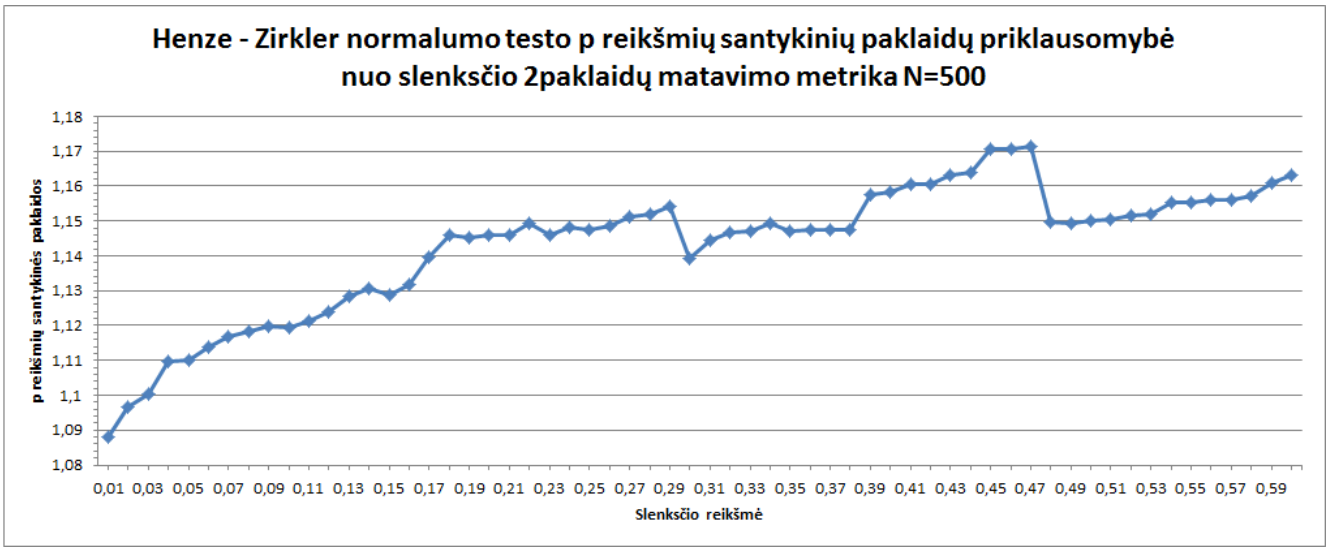
0.60 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas pirmąja metrika $N = 200$



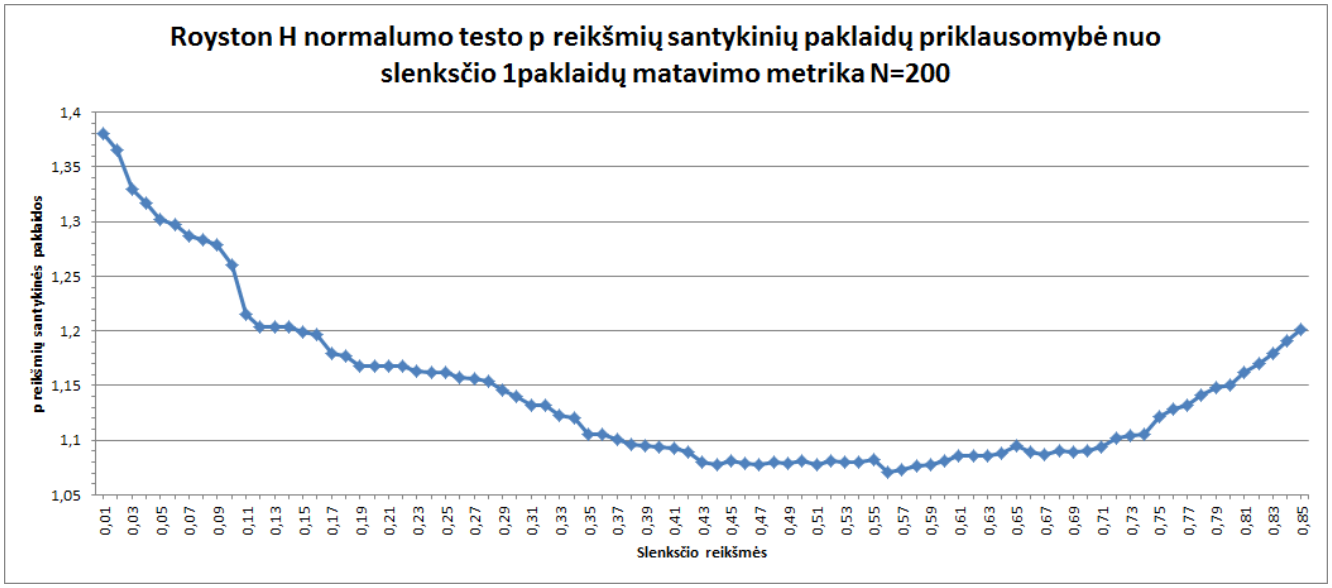
0.61 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas pirmąja metrika $N = 500$



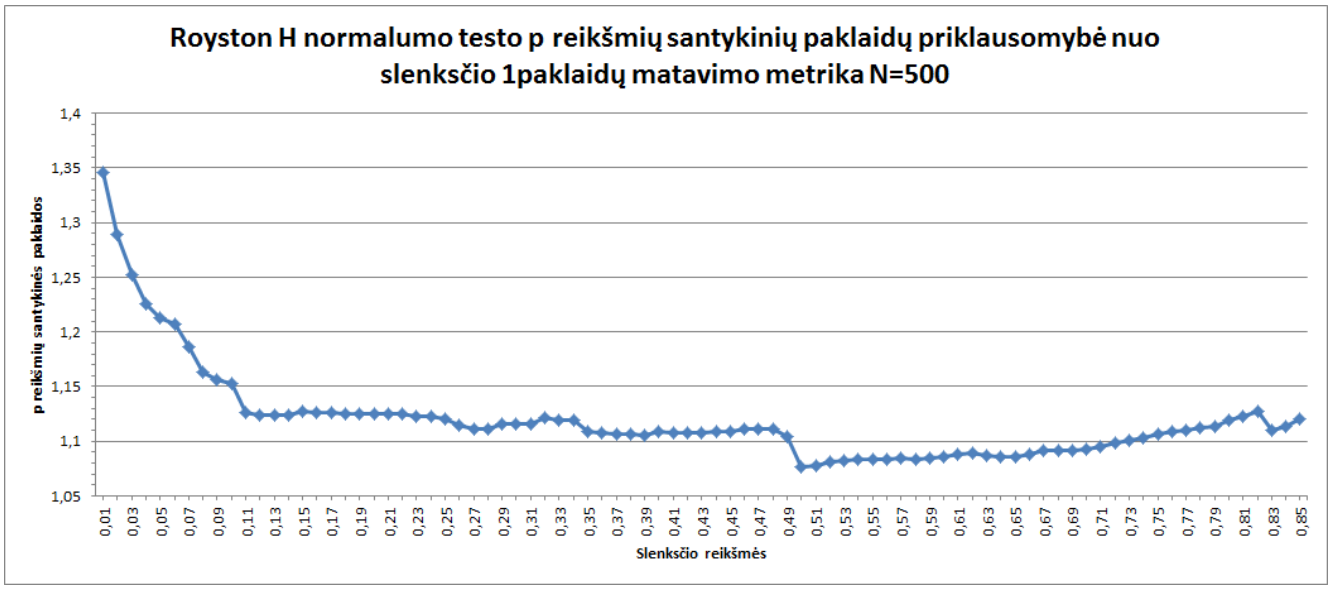
0.62 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas antrąja metrika $N = 200$



0.63 pav. Henze - Zirkler normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenkščio vertinant paklaidas antrąja metrika $N = 500$



0.64 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 200$



0.65 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas pirmąja metrika $N = 500$

0.66 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 200$

0.67 pav. Royston H normalumo testo p reikšmių santykinės paklaidos priklausomybė nuo slenksčio vertinant paklaidas antrąja metrika $N = 500$

PROGRAMOS TEKSAS

R failo konf_tyrimo_parametrai.r programos tekstas

```
# turiai <- c(100)
turiai <- c(100, 200, 500, 1000)
pol_eiles <- c(4, 6, 8)
# n_imciu <- 10
n_imciu <- 100
n_krypciu <- 25
turis_tikrinimo <- 5000
```

R failo konf_misiniu_parametrai.r programos tekstas

```
par_misiniu <- list(
  # R2
  # nepersidenge+
  '1_R2-nep'=list(
     list(svoris=3, mu=c(-2,1), sigma=array(c(1,0,0,2), c(2,2))),
     list(svoris=2, mu=c(-2,8), sigma=array(c(2,-1,-1,1), c(2,2))),
     list(svoris=3, mu=c(10,1), sigma=array(c(5,1,1,2), c(2,2))),
     list(svoris=2, mu=c(8,8), sigma=array(c(5,-2,-2,1), c(2,2)))),
  # vidutiniskai persidenge+
  '2_R2-vidp'=list(
    list(svoris=3, mu=c(17,5), sigma=array(c(2.5,1,1,2.1), c(2,2))),
    list(svoris=2, mu=c(-1,6), sigma=array(c(4,-1.7,-1.7,1), c(2,2))),
    list(svoris=3, mu=c(5,5), sigma=array(c(4,1,1,2), c(2,2))),
    list(svoris=2, mu=c(21,10), sigma=array(c(10,4.8,4.8,4), c(2,2)))),
  # stipriai persidenge+
  '3_R2-lbp'=list(
    list(svoris=3, mu=c(16,8), sigma=array(c(5,3,3,4), c(2,2))),
    list(svoris=2, mu=c(10,7), sigma=array(c(1.5,-0.4,-0.4,1), c(2,2))),
    list(svoris=3, mu=c(10,4), sigma=array(c(4,2,2,2), c(2,2))),
    list(svoris=2, mu=c(15,10), sigma=array(c(10,-4.8,-4.8,4), c(2,2)))),
  # R3
  # nepersidenge+
  '4_R3-nep'=list(
    list(svoris=3, mu=c(-2,1,4), sigma=array(c(1,0,0,0,2,0,0,0,3), c(3,3))),
    list(svoris=4, mu=c(-2,8,0), sigma=array(c(2,-1,0,-1,1,1,0,1,3), c(3,3))),
    list(svoris=2, mu=c(10,1,0), sigma=array(c(5,1,1,1,2,2.4,1,2.4,3), c(3,3))),
    list(svoris=2, mu=c(8,8,0), sigma=array(c(5,-2,0,-2,1,1,0,1,6), c(3,3)))),
  # vidutiniskai persidenge+
  '5_R3-vidp'=list(
    list(svoris=3, mu=c(17,5,0), sigma=array(c(2.5,1,0,1,2.1,1,0,1,1), c(3,3))),
    list(svoris=2, mu=c(-1,6,0), sigma=array(c(4,-1.7,0,-1.7,1,0.6,0,0.6,5),
c(3,3))),
    list(svoris=3, mu=c(5,5,0), sigma=array(c(4,1,1,1,2,2.8,1,2.8,4), c(3,3))),
    list(svoris=2, mu=c(21,10,0), sigma=array(c(10,4.8,0,4.8,4,1.2,0,1.2,1),
c(3,3)))),
  # stipriai persidenge+
  '6_R3-lbp'=list(
    list(svoris=3, mu=c(16,8,0), sigma=array(c(5,3,0,3,4,1,0,1,1), c(3,3))),
    list(svoris=2, mu=c(10,7,0), sigma=array(c( 1.5,-0.4,0,
                                               -0.4, 1,  1,
                                                  0, 1,  2), c(3,3))),
    list(svoris=3, mu=c(10,4,0), sigma=array(c(4  ,2  ,1.5,
                                               2  ,2  ,1.4,
                                               1.5,1.4,2  ), c(3,3))),
    list(svoris=2, mu=c(15,10,0), sigma=array(c( 10, -4.8,1,
                                                 -4.8, 4,  1.7,
                                                  1,   1.7,3  ), c(3,3)))),
  # R4
  # nepersidenge+
  '7_R4-nep'=list(
```

```
  list(svoris=3, mu=c(-2,1,4,0), sigma=array(c(1,0,0  ,0,
                                               0,2,0,  0,
                                               0,0,3,  1.5,
                                               0,0,1.5,2  ), c(4,4))),
  list(svoris=4, mu=c(-2,8,0,0), sigma=array(c( 2,-1,0,  0,
                                              -1, 1,1,  0,
                                               0, 1,3,  0.9,
                                               0, 0,0.9,1  ), c(4,4))),
  list(svoris=2, mu=c(10,1,0,0), sigma=array(c(5,1,  1,  0,
                                               1,2,  2.4,0,
                                               1,2.4,3,  0,
                                               0,0,  0,  3), c(4,4))),
  list(svoris=2, mu=c(8,8,0,0), sigma=array(c( 5,-2,0,  0,
                                             -2, 1,1,  0,
                                               0, 1,6,  0.5,
                                               0, 0,0.5,1  ), c(4,4)))),
# vidutiniskai persidenge+
'8_R4-vidp'=list(
   list(svoris=3, mu=c(17,5,0,0), sigma=array(c(2.5,1,  0,  0,
                                                1,  2.1,1,  0,
                                                0,  1,  1,  1.2,
                                                0,  0,  1.2,4  ), c(4,4))),
   list(svoris=2, mu=c(-1,6,0,0), sigma=array(c( 4,  -1.7,1,  0,
                                               -1.7, 1,  0.6,0,
                                                1,   0.6,5,  0,
                                                0,   0,  0,  1), c(4,4))),
   list(svoris=3, mu=c(5,5,0,0), sigma=array(c(4,1,  1,  0,
                                               1,2,  2.4,0,
                                               1,2.4,4,  0,
                                               0,0,  0,  3), c(4,4))),
   list(svoris=2, mu=c(21,10,0,0), sigma=array(c(10, 4.8,0,  1,
                                                 4.8,4,  1.2,1,
                                                 0,  1.2,1,  0.4,
                                                 1,  1,  0.4,1  ), c(4,4)))
),
# stipriai persidenge+
'9_R4-lbp'=list(
  list(svoris=3, mu=c(16,8,0,0), sigma=array(c(5,3,0,1,
                                               3,4,1,3.1,
                                               0,1,1,2.4,
                                               1,3.1,2.4,7), c(4,4))),
  list(svoris=2, mu=c(10,7,0,0), sigma=array(c(1.5,-0.4,0,1,
                                               -0.4,1,1,0,
                                               0,1,2,0,
                                               1,0,0,1), c(4,4))),
  list(svoris=3, mu=c(10,4,0,0), sigma=array(c(4,2,1.5,-1,
                                               2,2,1.4,-1,
                                               1.5,1.4,2,-1.4,
                                               -1,-1,-1.4,1), c(4,4))),
  list(svoris=2, mu=c(15,10,0,0), sigma=array(c(10,-4.8,1,0,
                                               -4.8,4,1.7,1,
                                               1,1.7,3,1,
                                               0,1,1,6), c(4,4)))),
#R5
#nepersidenge+
'10_R5-nep'=list(
  list(svoris=3, mu=c(-2,1,4,0,0), sigma=array(c(1,0,0,0,  0,
                                                 0,2,0,0,  0,
                                                 0,0,3,1.5,0,
                                                 0,0,1.5,1,0,
                                                 0,0,0,  0,1), c(5,5))),
  list(svoris=4, mu=c(-2,8,0,0,0), sigma=array(c( 2, -1,0,  0,  0,
                                                 -1,  1,1,  0,  0,
```

```
                                                    0,  1,3,  0.9,0,
                                                    0,  0,0.9,1,  0,
                                                    0,  0,0,  0,  1), c(5,5))),
    list(svoris=2, mu=c(10,1,0,0,0), sigma=array(c(5,1,  1,  0,0,
                                                   1,2,  2.4,0,0,
                                                   1,2.4,3,  0,0,
                                                   0,0,  0,  3,0,
                                                   0,0,  0,  0,1), c(5,5))),
    list(svoris=2, mu=c(8,8,0,0,0), sigma=array(c( 5,-2,0,  0,  0,
                                                  -2, 1,1,  0,  0,
                                                   0, 1,6,  0.5,0,
                                                   0, 0,0.5,1,  0,
                                                   0, 0,0,  0,  1), c(5,5)))),
  #vidutiniskai persidenge+
  '11_R5-vidp'=list(
      list(svoris=3, mu=c(17,5,0,0,0), sigma=array(c(2.5,1,  0,  0,  0,
                                                     1,  2.1,1,  0,  0,
                                                     0,  1,  1,  1.2,0,
                                                     0,  0,  1.2,4,  1,
                                                     0,  0,  0,  1,  3), c(5,5))),
      list(svoris=2, mu=c(-1,6,0,0,0), sigma=array(c( 4,  -1.7,1,  0,0,
                                                     -1.7, 1,  0.6,0,0,
                                                      1,   0.6,5,  0,0,
                                                      0,   0,  0,  1,0,
                                                      0,   0,  0,  0,2), c(5,5))),
      list(svoris=3, mu=c(5,5,0,0,0), sigma=array(c(4,1,  1,  0,0,
                                                    1,2,  2.4,0,0,
                                                    1,2.4,4,  0,0,
                                                    0,0,  0,  3,1.7,
                                                    0,0,  0,1.7,1  ), c(5,5))),
      list(svoris=2, mu=c(21,10,0,0,0), sigma=array(c(10, 4.8,0,  1,  0,
                                                      4.8,4,  1.2,1,  0,
                                                      0,  1.2,1,  0.4,0,
                                                      1,  1,  0.4,1,  1.2,
                                                      0,  0, 0,  1.2,3),c(5,5)))),
  # stipriai persidenge+
  '12_R5-lbp'=list(
    list(svoris=3, mu=c(16,8,0,0,0), sigma=array(c(5,3,  0,  1,  0,
                                                   3,4,  1,  3.1,0,
                                                   0,1,  1,  2.4,0,
                                                   1,3.1,2.4,6,  0.1,
                                                   0,0,  0,  0.1,5), c(5,5))),
    list(svoris=2, mu=c(10,7,0,0,0), sigma=array(c( 1.5,-0.4,0, 1,0,
                                                   -0.4, 1,  1,-1,0,
                                                    0,   1,  2,-1,0,
                                                    1,  -1, -1, 8,0,
                                                    0,   0,  0, 0,2), c(5,5))),
    list(svoris=3, mu=c(10,4,0,0,0), sigma=array(c( 4,   2,   1.5,-1,  0,
                                                    2,   2,   1.4,-1,  0,
                                                    1.5, 1.4, 2,  -1.4,0,
                                                   -1,  -1,  -1.4, 1,  0,
                                                    0,   0,   0,   0, 1),c(5,5))),
    list(svoris=2, mu=c(15,10,0,0,0), sigma=array(c( 10, -4.8,1,  0,  0,
                                                    -4.8, 4,  1.7,1,  0,
                                                     1,   1.7,3,  1,  0,
                                                     0,   1,  1,  6,  2.5,
                                                     0,   0, 0,  2.5,2),c(5,5)))))

persidengimo_pav <- list(
  'nep'='nepersidengę',
  'vidp'='vidutiniškai persidengę',
  'lbp'='gausiai persidengę')
```

R failo funk_tp_algoritmas.r programos tekstas

```
#-------------------------------------------------------------------------
# Pagalbines funkcijos
#-------------------------------------------------------------------------
normuoti_vektoriu <- function(x) {
  return(x / sqrt(sum(x ^ 2)))}
#-------------------------------------------------------------------------
# Imciu normavimas
#-------------------------------------------------------------------------
normuoti_pradine_imti <- function(imtis)
{
 tikr_skaida <- eigen(cov(imtis))
 vidurkiai <- colMeans(imtis)
 return(list(
   'vidurkiai' = vidurkiai,
   'tikr_vektoriai' = tikr_skaida$vectors,
   'tikr_reiksmes' = tikr_skaida$values,
   'imtis' = (imtis - matrix(
      vidurkiai, nrow(imtis), ncol=ncol(imtis), byrow = T)) %*%
        tikr_skaida$vectors %*% diag(1 / sqrt(tikr_skaida$values)))))}
normuoti_nauja_imti <- function(
  imtis, vidurkiai, tikr_vektoriai, tikr_reiksmes
) {
  return(
    (imtis - matrix(
      vidurkiai, nrow(imtis), ncol = ncol(imtis), byrow = T)) %*%
      tikr_vektoriai %*% diag(1 / sqrt(tikr_reiksmes)))}
#-------------------------------------------------------------------------
# Lezandro polinomu skaiciavimas
#-------------------------------------------------------------------------
gauti_polinomus <- function(imtis, eile)
{
  if (eile > 1) {
    lp <- cbind(rep(1, length(imtis)), imtis)
    for (j in 2:(eile)) {
      lp <- cbind(
         lp, ((2 * j - 1) * lp[, 2] * lp[, j] -
         (j - 1) * lp[, j - 1]) / j)
    }
  } else if (eile == 1) {
    lp <- cbind(rep(1, length(imtis)), imtis)
  } else {a
    lp <- matrix(1, nrow=length(imtis), ncol=1)
  }
  return(lp)
}
gauti_polinomu_isvestines <- function(lp)
{
  eile <- dim(lp)[2] - 1
  grad_lp <- cbind(rep(0, dim(lp)[1]), array(1, dim=(c(dim(lp)[1], 1))))
  if (eile > 1) {
    for (j in 2:eile) {
      grad_lp <- cbind(grad_lp, lp[, 2] * grad_lp[, j] + j * lp[, j])
    }
  } else {
    grad_lp <- grad_lp[, 1]
  }
  return(grad_lp)
}
#-------------------------------------------------------------------------
# Projektavimo indekso skaiciavimas
#-------------------------------------------------------------------------
gauti_indekso_reiksme <- function(kryptis, imtis, eile)
```

```
{
  kryptis <- normuoti_vektoriu(kryptis)
  lp <- gauti_polinomus(2 * pnorm(imtis %*% kryptis) - 1, eile)
  return(sum((colMeans(lp[, -1]) ** 2) * seq(3, by=2, length=eile)) / 2)
}
gauti_indekso_gradienta <- function(kryptis, imtis, eile)
{
    kryptis <- normuoti_vektoriu(kryptis)
    lc <- imtis %*% kryptis
    lp <- gauti_polinomus(2 * pnorm(imtis %*% kryptis) - 1, eile)
    isv_lp <- gauti_polinomu_isvestines(lp)

    gradientas <- vector("numeric", length(kryptis))
    for (k in 1:length(kryptis)) {
      gradientas[k] <- sqrt(2*pi) * (pi ** -1) * sum(
        colMeans(lp[, -1]) *
          colMeans(
            isv_lp[, -1] * array(exp(-(lc ** 2) / 2), dim=dim(isv_lp[, -1])) *
              array((imtis[, k] - kryptis[k] * lc), dim=dim(isv_lp[, -1])) *
          seq(3, by=2, length=eile))
    }
    return(gradientas)
}
#-------------------------------------------------------------------------------
# Randama pradine kryptis optimizavimui - gradientiniam nusileidimui
#-------------------------------------------------------------------------------
gauti_pirmine_krypti <- function(imtis, eile, max_iter=1000)
{
  dim <- ncol(imtis)
  asis <- c(1, rep(0, dim - 1))
  max_indeksas <- gauti_indekso_reiksme(asis, imtis, eile)
  idomiausia_kryptis <- asis
  for (i in 2:dim) {
    asis[i] <- 1
    asis[i-1] <- 0
    indeksas <- gauti_indekso_reiksme(asis, imtis, eile)
    if (indeksas > max_indeksas) {
      max_indeksas <- indeksas
      idomiausia_kryptis <- asis
    }}
  for (iter in 1:max_iter) {
    indeksas <- max_indeksas
    for (i in 1:dim) {
      asis <- rep(0, dim)
      asis[i] <- 1
      ilgis <- sqrt(sum((idomiausia_kryptis + asis) ^ 2))
      if (ilgis > 1e-3) {
        nauja_kryptis <- (idomiausia_kryptis + asis) / ilgis
        naujas_indeksas <- gauti_indekso_reiksme(nauja_kryptis, imtis, eile)
        if (naujas_indeksas > indeksas) {
          indeksas <- naujas_indeksas
          kryptis <- nauja_kryptis
        } }
      ilgis <- sqrt(sum((idomiausia_kryptis - asis) ^ 2))
      if (ilgis > 1e-3) {
        nauja_kryptis <- (idomiausia_kryptis - asis) / ilgis
        naujas_indeksas <- gauti_indekso_reiksme(nauja_kryptis, imtis, eile)
        if (naujas_indeksas > indeksas) {
          indeksas <- naujas_indeksas
          kryptis <- nauja_kryptis
        }}}
    if (max_indeksas == indeksas) {
      break
```

```
    }
    max_indeksas <- indeksas
    idomiausia_kryptis <- kryptis
    iter <- iter + 1
  }
  #print(kryptis)
  return(idomiausia_kryptis)
}
#-------------------------------------------------------------------------------
# Projektavimo krypties, maksimizuojancios projektavimo indeksa, radimas
#-------------------------------------------------------------------------------
gauti_optimalia_krypti <- function(imtis, pradine_kryptis, eile)
{
  optim_rez <- optim(par=pradine_kryptis,
                     fn=gauti_indekso_reiksme, gr=gauti_indekso_gradienta,
                     imtis, eile,
                     method='BFGS', control=list(fnscale=-1))
  return(list(kryptis=normuoti_vektoriu(optim_rez$par),
              indeksas=optim_rez$value))
}
#-------------------------------------------------------------------------------
# Strukturos panakinimas
#-------------------------------------------------------------------------------
panaikinti_struktura <- function(imtis, kryptis)
{
  gausanizuota_projekcija <- qnorm(
    rank(imtis %*% kryptis) / length(imtis %*% kryptis) -
      (2 * length(imtis %*% kryptis)) ** -1)
  return(imtis - (imtis %*% kryptis) %*% t(kryptis) +
           gausanizuota_projekcija %*% t(kryptis))
}
#-------------------------------------------------------------------------------
# Tikslinis projektavimas
#-------------------------------------------------------------------------------
atlikti_tikslini_projektavima <- function(
  n_krypciu, imtis, eile, ar_kaupti_imtis=FALSE
) {
  kryptys <- list()
  indeksai <- list()
  imtys <- list()
  for (i in 1:n_krypciu) {
    if (ar_kaupti_imtis == TRUE || i == 1) {
      imtys[[i]] <- imtis
    }
    optim_rezultatas <- gauti_optimalia_krypti(
      imtis, gauti_pirmine_krypti(imtis, eile), eile)
    imtis <- panaikinti_struktura(imtis, optim_rezultatas$kryptis)
    kryptys[[i]] <- optim_rezultatas$kryptis
    indeksai[[i]] <- optim_rezultatas$indeksas
  }
  return(list(kryptys=kryptys, indeksai=indeksai, imtys=imtys))
}
#-------------------------------------------------------------------------------
# Imciu su panaikinta struktura gavimas is pradines imties tankio vertinimui
#-------------------------------------------------------------------------------
gauti_tarpines_imtis <- function(kryptys, imtis)
{
  imtys <- list()
  for (i in 1:length(kryptys)) {
    imtys[[i]] <- imtis
    imtis <- panaikinti_struktura(imtis, kryptys[[i]])
  }
  return(imtys)
```

```
}
#-------------------------------------------------------------------------------
# Tankio vertinimas naudojant tikslinio projektavimo rezultatus
#-------------------------------------------------------------------------------
gauti_tankio_iverti <- function(
  nauja_imtis, tp_imtys, tp_kryptys, n_krypciu=NULL, eile
) {
  require(package = 'mvtnorm', quietly=TRUE)
  sandauga <- 1
  if (is.null(n_krypciu) || n_krypciu > length(tp_kryptys)) {
    n_krypciu <- length(tp_kryptys)
  }
  if (n_krypciu >= 1) {
    for (k in 1:n_krypciu) {
      proj <- nauja_imtis %*% tp_kryptys[[k]]
      tp_proj <- tp_imtys[[k]] %*% tp_kryptys[[k]]

      lp <- gauti_polinomus(2 * pnorm(proj) - 1, eile)
      lp_koef <- gauti_polinomus(2 * pnorm(tp_proj) - 1, eile)

      koef <- colMeans(lp_koef) * seq(1, by=2, length=eile + 1)
      proj_koef <- lp %*% koef
      sandauga <- sandauga * proj_koef
    }}
  sandauga[sandauga < 0] <- 0
  return(dmvnorm(nauja_imtis) * sandauga)}
```

R failo funk_gauso_misinys.r programos tekstas

```
#-------------------------------------------------------------------------------
# Gauso misinio funkcijos
#-------------------------------------------------------------------------------
library('mvtnorm')
gauti_misinio_tanki <- function (x, par_misinio) {
  svoriu_suma <- gauti_misinio_svoriu_suma(par_misinio)
  misinio_tankis <- 0
  for (par_skirstinio in par_misinio) {
    misinio_tankis <- misinio_tankis +
      dmvnorm(
        x=x, mean=par_skirstinio[[2]], sigma=par_skirstinio[[3]]) *
      par_skirstinio[[1]] / svoriu_suma
  }
  return(misinio_tankis)
}
generuoti_misini <- function (
  turis, par_misinio, seed=NULL,
  ar_apjungti_skirstinius=TRUE
) {
  svoriu_suma <- gauti_misinio_svoriu_suma(par_misinio)
  if (!is.null(seed)) {
    set.seed(seed)
  }
  skirstiniai <- lapply(
    X = par_misinio,
    FUN = function(par_skirstinio) {
      rmvnorm(
        n=gauti_skirstinio_turi(par_skirstinio[[1]], svoriu_suma, turis),
        mean=par_skirstinio[[2]],
        sigma=par_skirstinio[[3]])
    })
    if (ar_apjungti_skirstinius) {
    res <- do.call(rbind, skirstiniai)
  } else {
    res <- skirstiniai
```

```
  }
  return(res)
}
#-------------------------------------------------------------------------------
# Pagalbines Gauso misinio funkcijos
#-------------------------------------------------------------------------------
gauti_skirstinio_turi <- function(
  skirstinio_svoris, svoriu_suma, misinio_turis
) {
  return(round(skirstinio_svoris / svoriu_suma * misinio_turis, 0))
}
gauti_misinio_svoriu_suma <- function(par_skirstiniu) {
  return(sum(sapply(par_skirstiniu, function(x) x[[1]])))
}
```

R failo funk_paklaidos.r programos tekstas

```
gauti_pakl1 <- function(tankis, tankioivertis)
{
  # argumentai: vektorius, vektorius
  # grazinama: skaicius
  return(mean((tankioivertis - tankis) ** 2))
}
gauti_pakl2 <- function(tankis, tankioivertis)
{
  # argumentai: vektorius, vektorius
  # grazinama: skaicius
  return(mean(((tankioivertis - tankis) ** 2) / tankis))
}
```

R failo funk_grafikai.r programos tekstas

```
source('konf_tyrimo_parametrai.r')
source('funk_paklaidos.r')
source('funk_grafiku_antrastes.r')
#sriftas <- 'sans'  # Arial
sriftas <- 'serif'  # Times
n_padal_x <- 10
n_padal_y <- 6
lin_plot <- 0.7
y_tikslumas <- 3
# 0 => balta, 1 => juoda
pi_sviesumas <- 0.1
slekscio_storis <- 0.5
ar_pi_spalvoti <- FALSE
################################################################################
# Pagalbinės funkcijos
################################################################################
gg_color_hue <- function(n) {
  hues = seq(15, 375, length=n+1)
  return(hcl(h=hues, l=65, c=100)[1:n])
}
as.numeric.factor <- function(x) {
  return(as.numeric(levels(x))[x])
}
gauti_duomenis_grafikui <- function(df_duomenys, n_krypciu, pol_eiles=NULL)
{
  rez <- df_duomenys[df_duomenys$K <= n_krypciu, ]
  if (! is.null(pol_eiles)) {
    rez <- rez[rez$J %in% pol_eiles, ]
  }
  return(rez)
}
sukurti_filtra <- function(filtro_reiksmes, filtruojamos_reiksmes)
```

```
{
  return(function() {
    if (! is.null(filtro_reiksmes)) {
      rez <- intersect(filtro_reiksmes, filtruojamos_reiksmes)
    } else {
      rez <- filtruojamos_reiksmes
    }
    return(rez)
  })}
##############################################################################
graf_paklaidos <- function(duomenys, i_pakl, ar_su_pi=FALSE)
{
  require('ggplot2', quietly=TRUE)
  if (ar_su_pi == TRUE) {
    min_y <- min(duomenys$ci_lower, na.rm=TRUE)
    min_y <- ifelse(min_y < 0, 0, min_y)
    max_y <- max(duomenys$ci_upper, na.rm=TRUE)
  } else {
    min_y <- min(duomenys$vid_reiksme, na.rm=TRUE)
    max_y <- max(duomenys$vid_reiksme, na.rm=TRUE)
  }
  graf <- graf_bazinis(duomenys, ar_su_pi, 0) +
    scale_y_continuous(breaks=round(seq(min_y, max_y, length=n_padal_y),
                                    y_tikslumas)) +
    labs(title=pav_pakl(i_pakl, turis_tikrinimo)) +
    ylab(ylab_pakl(i_pakl, n_imciu, TRUE))
  return(graf)
}
graf_indeksai <- function(duomenys, ar_su_pi=FALSE,
                          slenkstis_1=NULL, slenkstis_2=NULL)
{
  require('ggplot2', quietly=TRUE)
  if (ar_su_pi == TRUE) {
    min_y <- min(duomenys$ci_lower, na.rm=TRUE)
    min_y <- ifelse(min_y < 0, 0, min_y)
    max_y <- max(duomenys$ci_upper, na.rm=TRUE)
  } else {
    min_y <- min(duomenys$vid_reiksme, na.rm=TRUE)
    max_y <- max(duomenys$vid_reiksme, na.rm=TRUE)
  }
  graf <- graf_bazinis(duomenys, ar_su_pi, 1) +
    scale_y_continuous(breaks=round(seq(min_y, max_y, length=n_padal_y),
                                    y_tikslumas)) +
    labs(title=pav_indeksas(FALSE)) +
    ylab(ylab_indeksas())
  if (! is.null(slenkstis_1)) {
    graf <- graf +
        geom_hline(aes(yintercept=slenkstis_1), colour='red',
                   size=slekscio_storis)
  }
  if (! is.null(slenkstis_2)) {
    graf <- graf +
      geom_hline(aes(yintercept=slenkstis_2), colour='blue',
                 size=slekscio_storis)
  }
  return(graf)
}
graf_indeksai_2 <- function(duomenys, dim, ar_su_pi=FALSE,
                            slenkstis_1=NULL, slenkstis_2=NULL)
{
  require('ggplot2', quietly=TRUE)
  if (ar_su_pi == TRUE) {
    min_y <- min(duomenys$ci_lower, na.rm=TRUE)
```

```
    min_y <- ifelse(min_y < 0, 0, min_y)
    max_y <- max(duomenys$ci_upper, na.rm=TRUE)
  } else {
    min_y <- min(duomenys$vid_reiksme, na.rm=TRUE)
    max_y <- max(duomenys$vid_reiksme, na.rm=TRUE)
  }
  graf <- graf_bazinis(duomenys, ar_su_pi, 1) +
    scale_y_continuous(breaks=round(seq(min_y, max_y, length=n_padal_y),
                                    y_tikslumas)) +
    labs(title=bquote(Y %in% R^.(dim))) +
    ylab(ylab_indeksas())
  if (! is.null(slenkstis_1)) {
    graf <- graf +
      geom_hline(aes(yintercept=slenkstis_1), colour='red',
                 size=slekscio_storis)
  }
  if (! is.null(slenkstis_2)) {
    graf <- graf +
      geom_hline(aes(yintercept=slenkstis_2), colour='blue',
                 size=slekscio_storis)
      }
  return(graf)
}
sukurti_graf_mvnorm_test <- function(pav)
{
  return(function(duomenys, ar_su_pi=FALSE)
  {
    require('ggplot2', quietly=TRUE)
    duomenys$K <- duomenys$K - 1
    graf <- graf_bazinis(duomenys, ar_su_pi, 0) +
      labs(title=pav) +
      geom_hline(aes(yintercept=0.05), colour='red', size=slekscio_storis) +
      scale_y_continuous(limits=c(0, 1),
                         breaks=round(seq(0, 1, length=n_padal_y),
                                      y_tikslumas)) +
      ylab('p')
    return(graf)
  })}
graf_bazinis <- function(duomenys, ar_su_pi=FALSE, x_pradzia=1)
{
  require('ggplot2', quietly=TRUE)
  n_K <- length(unique(duomenys$K))
  graf <- ggplot(duomenys, aes(x=K, y=vid_reiksme, group=J)) +
      geom_line(aes(color=J), size = lin_plot) +
      xlab("K") +
      scale_x_discrete(breaks=unique(
        round(seq(x_pradzia, n_K, length=n_padal_x)))) +
      theme_bw() +
      theme(plot.title=element_text(family=sriftas, size=12)) +
      theme(axis.title.x=element_text(family=sriftas, size=10)) +
      theme(axis.title.y=element_text(family=sriftas, size=10))
  pi_aes_arg <- list(ymin='ci_lower',ymax='ci_upper')
  if (ar_pi_spalvoti == TRUE) {
    pi_aes_arg$fill = 'J'  }
  if (ar_su_pi == TRUE) {
    graf <- graf + geom_ribbon(data=duomenys,
                               do.call(aes_string, pi_aes_arg),
                               alpha=pi_sviesumas)  }
  spalvos <- gg_color_hue(length(pol_eiles))[
    match(unique(as.numeric.factor(duomenys$J)), pol_eiles)]
  spalvos <- c(spalvos, '#000000')
  graf <- graf +
  scale_colour_grey(start = 0, end = .8) +
```

```
  scale_fill_grey(start = 0, end = .8)
  return(graf)
}
```

R failo funk_grafiku_antrastes.r programos tekstas

```
source('konf_tyrimo_parametrai.r')
pav_pakl <- function(i_pakl, turis) {
  pavadinimai <- list(
    bquote(sigma[1] ~ '=' ~ frac(1, .(turis)) ~ sum(bgroup(
      "(", f(x[i]) - widehat(f)(x[i]), ")") ^ 2, i == 1, .(turis))),
    bquote(sigma[2] ~ '=' ~ frac(1, .(turis)) ~ sum(
      frac(group("(", f(x[i]) - widehat(f)(x[i]), ")") ^ 2,
           f(x[i])), i == 1, .(turis)))
  )
  return(pavadinimai[[i_pakl]])
}
ylab_pakl <- function(i_pakl, n_imciu, ar_norm) {
  if (ar_norm == TRUE) {
    ylab <- bquote(frac(1, .(n_imciu)) * sum(sigma[.(i_pakl)] * "'" * (K)))
  } else {
    ylab <- bquote(frac(1, .(n_imciu)) ~ sum(bgroup(
      sigma[.(i_pakl)]^(K) )))
  }
  return(ylab)
}
pav_indeksas <- function(is_full=FALSE) {
  if (is_full) {
    arg_lp <- substitute(2 * Phi(alpha^{T} * z[i]) - 1)
  } else {
    arg_lp <- substitute(R[i])
  }
  expr <- substitute(
    widehat(I)(alpha) == frac(1, 2) * ' ' * sum(
      (2 * j + 1) * ' ' * frac(1, N) * ' ' * bgroup(
        '[', sum(P[j](arg_lp), i==1, N), ']')^{2},
      j==1, J), list(arg_lp=arg_lp))
  return(expr)
}
ylab_indeksas <- function() {
  expr <- bquote(frac(1, .(n_imciu)) * ' ' * sum(widehat(I))(alpha^{(K)}))
  return(expr)
}
```

R failo funk_2d_tankis.r programos tekstas

```
gauti_tinkleli <- function(x=(0:10)/10, y=x)
{
  n_x <- length(x)
  n_y <- length(y)
  xoy <- cbind(rep(x, n_y), as.vector(matrix(y, n_x, n_y, byrow=TRUE)))
  XY <- matrix(xoy, n_x * n_y, 2, byrow=FALSE)
  return(XY)
}
gauti_xy_asis <- function(duomenys_2d, n_tasku_asyje=200)
{
  reziai <- apply(X=duomenys_2d, MARGIN=2, FUN=range)
  return(list(x=seq(reziai[1, 1], reziai[2, 1], len=n_tasku_asyje),
              y=seq(reziai[1, 2], reziai[2, 2], len=n_tasku_asyje)))
}
```

R failo vykd_tikslinis_projektavimas.r programos tekstas

```
source('funk_tp_algoritmas.r')
source('funk_gauso_misinys.r')
```

```
source('konf_misiniu_parametrai.r')
source('konf_tyrimo_parametrai.r')
s <- function(x) {
  return(as.character(x))
}
seed <- 1
rez_tps <- list()
for (i_misinio in names(par_misiniu)) {
  rez_tps[[s(i_misinio)]] <- list()
  for (turis in turiai) {
    rez_tps[[s(i_misinio)]][[s(turis)]] <- list()
    for (eile in pol_eiles) {
      rez_tps[[s(i_misinio)]][[s(turis)]][[s(eile)]] <- list()
      for (i_imties in 1:n_imciu) {
        rez_tps[[s(i_misinio)]][[s(turis)]][[s(eile)]][[i_imties]] <- list()
        set.seed(seed)
        imtis <- generuoti_misini(turis, par_misiniu[[i_misinio]], seed,
                                  ar_apjungti_skirstinius=TRUE)
        seed <- seed + 1
                rez_norm <- normuoti_pradine_imti(imtis)
        rez_tp <- atlikti_tikslini_projektavima(n_krypciu, rez_norm$imtis, eile)
        rez_norm <- rez_norm[names(rez_norm) != 'imtis']
        rez_tps[[s(i_misinio)]][[s(turis)]][[s(eile)]][[i_imties]]$sf <- rez_norm
        rez_tps[[s(i_misinio)]][[s(turis)]][[s(eile)]][[i_imties]]$tp <- rez_tp
        print(paste(format(Sys.time(), "%Y-%m-%d %H:%M:%S     "),
                    "Misinys ", i_misinio, ", N = ", turis,
                    ", J = ", eile, ", imtis ", i_imties, sep=""))
      }}}}
save(rez_tps, file = paste0("tp_rezultatai_",
                            ".RData"))
```

R failo vykd_tikslumo tyrimas.r programos tekstas

```
source('funk_tp_algoritmas.r')
source('funk_gauso_misinys.r')
source('funk_paklaidos.r')
source('konf_misiniu_parametrai.r')
source('konf_tyrimo_parametrai.r')
require('MVN')
load('tp_rezultatai_.RData')
seed <- 1
rez_graf <- list()
for (i_misinio in names(rez_tps)) {
  rez_graf[[i_misinio]] <- list()
  for (turis in names(rez_tps[[i_misinio]])) {
    # Paruošiamos lentelės suvidurkintoms reikšmėms
    rez_graf[[i_misinio]][[turis]] <- list(
      df_indeksai=data.frame(),
      df_paklaidos1=data.frame(),
      df_paklaidos2=data.frame(),
      df_paklaidos1norm=data.frame(),
      df_paklaidos2norm=data.frame(),
      df_mardia_skew=data.frame(),
      df_mardia_kurt=data.frame(),
      df_hz=data.frame(),
      df_royston=data.frame())
    for (eile in names(rez_tps[[i_misinio]][[turis]])) {
      df_imciu <- data.frame()
      for (i_imties in 1:length(rez_tps[[i_misinio]][[turis]][[eile]])) {
        set.seed(seed)
        imtis <- generuoti_misini(
          turis_tikrinimo, par_misiniu[[i_misinio]], seed, TRUE)
        seed <- seed + 1
        imtis_norm <- normuoti_nauja_imti(
```

```
    imtis,
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$sf$vidurkiai,
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$sf$tikr_vektoriai,
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$sf$tikr_reiksmes
  )
  print(paste0(format(Sys.time(), "%Y-%m-%d %H:%M:%S     "),
               "Misinys ", i_misinio, ", N = ", turis,
               ", J = ", eile, ", imtis ", i_imties))
  tankis <- gauti_misinio_tanki(imtis, par_misiniu[[i_misinio]])
  tankis_be_krypciu <- dmvnorm(imtis_norm) / sqrt(prod(
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$sf$tikr_reiksmes))
  norm_konst1 <- gauti_pakl1(tankis, tankis_be_krypciu)
  norm_konst2 <- gauti_pakl2(tankis, tankis_be_krypciu)
  df_imciu <- rbind(df_imciu, data.frame(
    i_imties=i_imties,
    pol_eile=eile,
    n_krypciu=0,
    pakl1=norm_konst1,
    pakl2=norm_konst2,
    pakl1norm=1,
    pakl2norm=1,
    indekso_reiksme=NA,
    p_mardia_skew=NA,
    p_mardia_kurt=NA,
    p_hz=NA,
    p_royston=NA))
  tarpines_imtys <- gauti_tarpines_imtis(
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$tp$kryptys,
    rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$tp$imtys[[1]])
  for (k in 1:n_krypciu) {
    tankio_ivertis <- gauti_tankio_iverti(
      imtis_norm,
      tarpines_imtys,
      rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$tp$kryptys,
      k,
      as.numeric(eile))
    tankio_ivertis <- tankio_ivertis / sqrt(prod(
      rez_tps[[i_misinio]][[turis]][[eile]][[i_imties]]$sf$tikr_reiksmes))
    pakl1 <- gauti_pakl1(tankis, tankio_ivertis)
    pakl2 <- gauti_pakl2(tankis, tankio_ivertis)
    rez_mardia <- mardiaTest(tarpines_imtys[[k]])
    rez_hz <- hzTest(tarpines_imtys[[k]])
    rez_royston <- roystonTest(tarpines_imtys[[k]])
    df_imciu <- rbind(df_imciu, data.frame(
      i_imties=i_imties,
      pol_eile=eile,
      n_krypciu=k,
      pakl1=pakl1,
      pakl2=pakl2,
      pakl1norm=pakl1 / norm_konst1,
      pakl2norm=pakl2 / norm_konst2,
      indekso_reiksme=rez_tps[[i_misinio]][[turis]][[eile]][[
        i_imties]]$tp$indeksai[[k]],
      p_mardia_skew= rez_mardia@p.value.skew,
      p_mardia_kurt=rez_mardia@p.value.kurt,
      p_hz=rez_hz@p.value,
      p_royston=rez_royston@p.value))
  }}
df_be_krypciu <- df_imciu[df_imciu$n_krypciu == 0, ]
dfs_rysiai <- list(
  df_paklaidos1='pakl1',
  df_paklaidos2='pakl2',
  df_paklaidos1norm='pakl1norm',
```

```
        df_paklaidos2norm='pakl2norm',
        df_indeksai='indekso_reiksme',
        df_mardia_skew='p_mardia_skew',
        df_mardia_kurt='p_mardia_kurt',
        df_hz='p_hz',
        df_royston='p_royston')
      for (df_pav in c('df_paklaidos1', 'df_paklaidos2',
                       'df_paklaidos1norm', 'df_paklaidos2norm')) {
        rez_eilute <- list(K=0, J=eile)
        if (df_pav %in% c('df_paklaidos1norm', 'df_paklaidos2norm')) {
          rez_eilute$vid_reiksme <- 1
          rez_eilute$ci_lower <- 1
          rez_eilute$ci_upper <- 1
        } else {
          n_x <- length(df_be_krypciu[[dfs_rysiai[[df_pav]]]])
          var_x <- var(df_be_krypciu[[dfs_rysiai[[df_pav]]]])
          vid_x <- mean(df_be_krypciu[[dfs_rysiai[[df_pav]]]])
          if (sqrt(var_x / n_x) > 10 * .Machine$double.eps * abs(vid_x)) {
            rez_t_test <- t.test(df_be_krypciu[[dfs_rysiai[[df_pav]]]])
          } else {
            rez_t_test <- list(
              estimate=vid_x,
              conf.int=rep(vid_x, 2))
          }
          rez_eilute$vid_reiksme <- rez_t_test$estimate
          rez_eilute$ci_lower <- rez_t_test$conf.int[1]
          rez_eilute$ci_upper <- rez_t_test$conf.int[2]
        }
        rez_graf[[i_misinio]][[turis]][[df_pav]] <- rbind(
          rez_graf[[i_misinio]][[turis]][[df_pav]],
          data.frame(rez_eilute))
      }
      for (k in 1:n_krypciu) {
        df_k_krypties <- df_imciu[df_imciu$n_krypciu == k, ]
        for (df_pav in names(dfs_rysiai)) {
          n_x <- length(df_k_krypties[[dfs_rysiai[[df_pav]]]])
          var_x <- var(df_k_krypties[[dfs_rysiai[[df_pav]]]])
          vid_x <- mean(df_k_krypties[[dfs_rysiai[[df_pav]]]])
          if (sqrt(var_x / n_x) > 10 * .Machine$double.eps * abs(vid_x)) {
            rez_t_test <- t.test(df_k_krypties[[dfs_rysiai[[df_pav]]]])
          } else {
            rez_t_test <- list(
              estimate=vid_x,
              conf.int=rep(vid_x, 2))
          }
          rez_graf[[i_misinio]][[turis]][[df_pav]] <- rbind(
            rez_graf[[i_misinio]][[turis]][[df_pav]],
            data.frame(
              K=k,
              J=eile,
              vid_reiksme=rez_t_test$estimate,
              ci_lower=rez_t_test$conf.int[1],
              ci_upper=rez_t_test$conf.int[2]))
        }}}
    for (df_pav in names(rez_graf[[i_misinio]][[turis]])) {
      rez_graf[[i_misinio]][[turis]][[df_pav]]$J <- as.factor(
        rez_graf[[i_misinio]][[turis]][[df_pav]]$J)
    }
    print(paste0(format(Sys.time(), "%Y-%m-%d %H:%M:%S                "),
                 "Misinys ", i_misinio, ", N = ", turis,
                 "        Suvidurkinta"))
  }}
save(rez_graf, file = paste0("tp_paklaidos_",
```

```
                                ".RData"))
```

R failo vykd_tyrimo_grafikai.r programos tekstas

```
require("ggplot2")
require("gridExtra")
source("funk_grafiku_antrastes.r")
source("konf_tyrimo_parametrai.r")
source.with.encoding('konf_misiniu_parametrai.r', encoding='utf-8')
source('funk_grafikai.r')
load(file="tp_paklaidos_.RData")
### rezultatu filtravimo parametrai; NULL - nefiltruoti
filt_misiniai <- NULL
# filt_misiniai <- c('12_R5-lbp')
filt_turiai <- NULL
# filt_turiai <- c('1000')
 filt_eiles <- NULL
# filt_eiles <- c(6, 8)
### Grafiku braizymo parametrai
n_krypciu_grafikui <- 100
ar_braizyti_pi <- TRUE
slenkstis_1 <- NULL
slenkstis_2 <- NULL
f_misiniai <- sukurti_filtra(filt_misiniai, names(rez_graf))
for (i_misinio in f_misiniai()) {
  dim <- length(par_misiniu[[i_misinio]][[1]][[2]])
  f_turiai <- sukurti_filtra(filt_turiai, names(rez_graf[[i_misinio]]))
  for (tur in f_turiai()) {
    pav <- bquote(
      Y %in% R^.(dim) * ',' ~
        .(persidengimo_pav[[strsplit(i_misinio, '-')[[1]][2]]]) * ',' ~
        tiksliniam ~ projektavimui ~ N ~ '=' ~ .(tur))
    grafikai1 <- list()
    for (i_pakl in 1:2) {
      grafikai1[[i_pakl]] <- graf_paklaidos(
        gauti_duomenis_grafikui(
          rez_graf[[i_misinio]][[tur]][[paste0('df_paklaidos',
                                               i_pakl, 'norm')]],
          n_krypciu_grafikui, filt_eiles),
        i_pakl, ar_braizyti_pi)
    }
    grafikai1[[3]] <- graf_indeksai(
      gauti_duomenis_grafikui(
        rez_graf[[i_misinio]][[tur]]$df_indeksai,
        n_krypciu_grafikui, filt_eiles),
      ar_braizyti_pi, slenkstis_1, slenkstis_2)
    normalumo_grafiku_funkcijos <- list(
      sukurti_graf_mvnorm_test('Mardia: asimetrija'),
      sukurti_graf_mvnorm_test('Mardia: ekscesas'),
      sukurti_graf_mvnorm_test('Henze-Zirkler'),
      sukurti_graf_mvnorm_test('Royston'))
    normalumo_grafiku_df_pavadinimai <- c(
      'df_mardia_skew',
      'df_mardia_kurt',
      'df_hz',
      'df_royston')
    grafikai2 <- mapply(
      function(graf_fn, df_pav, filt_eiles, ar_braizyti_pi) {
        graf_fn(
          gauti_duomenis_grafikui(
            rez_graf[[i_misinio]][[tur]][[df_pav]],
            n_krypciu_grafikui, filt_eiles), ar_braizyti_pi)
      },
      graf_fn=normalumo_grafiku_funkcijos,
```

```
      df_pav=normalumo_grafiku_df_pavadinimai,
      MoreArgs=list(filt_eiles=filt_eiles), ar_braizyti_pi=ar_braizyti_pi,
      SIMPLIFY=FALSE)
    grafikas1 <- do.call(marrangeGrob, c(grafikai1, list(
      nrow=2, ncol=2,
      top=textGrob(pav, gp=gpar(fontfamily=sriftas, fontsize=12)))))
    grafikas2 <- do.call(marrangeGrob, c(grafikai2, list(
      nrow=2, ncol=2,
      top=textGrob(pav, gp=gpar(fontfamily=sriftas, fontsize=12)))))
    failo_pav <- paste0(
      'mis', i_misinio, "_n", tur, '_im', n_imciu,
      ifelse(! is.null(filt_eiles),
             paste0('_j', paste0(filt_eiles, collapse='-')), ''),
      ifelse(ar_braizyti_pi, '_pi', ''))
    ggsave(filename=paste0(
      failo_pav, '_1.png'),
      path='grafikai/',
      dpi=100,
      plot=grafikas1)
    ggsave(filename=paste0(
      failo_pav, '_2.png'),
      path='grafikai/',
      dpi=100,
      plot=grafikas2)
  }}
```

R failo vykd_misiniu_sklaidos_diagramos.r programos tekstas

```
source.with.encoding('konf_misiniu_parametrai.r', encoding='utf-8')
source('funk_gauso_misinys.r')
source('funk_tp_algoritmas.r')
# http://www.inside-r.org/packages/cran/GGally/docs/ggpairs
# http://www.inside-r.org/packages/cran/GGally/docs
# http://cran.at.r-project.org/web/packages/GGally/GGally.pdf
library('GGally')
ar_spausdinti_pradini <- TRUE
ar_spausdinti_normuota <- TRUE
seed <- 1000
turis <- 1000
for (i_misinio in names(par_misiniu)) {
  skirstiniai <- generuoti_misini(turis, par_misiniu[[i_misinio]], seed, FALSE)
  for (j in 1:length(skirstiniai)) {
    skirstiniai[[j]] <- data.frame(skirstiniai[[j]])
    skirstiniai[[j]] <- cbind(skirstiniai[[j]], klase=factor(j))
  }
  misinys <- do.call(rbind, skirstiniai)
  pav <- bquote(
    Y %in% R^.(length(par_misiniu[[i_misinio]][[1]][[2]])) * ',' ~
      .(persidengimo_pav[[strsplit(i_misinio, '-')[[1]][2]]]) * ',' ~
       N ~ '=' ~ .(turis))
  if (ar_spausdinti_pradini) {
    plot <- ggpairs(
      misinys, colour='klase', alpha=0.4,
      upper=list(continuous = 'points', combo = 'box'),
      lower=list(continuous = 'density', combo = 'facethist'),
      diag=list(continuous='density', discrete='bar'),
      axisLabels="show",
      title=bquote('Pradinis mišinys' ~ .(pav))
    )
    print(plot)
  }
  if (ar_spausdinti_normuota) {
    norm_duomenys <- normuoti_pradine_imti(as.matrix(misinys[, -ncol(misinys)]))
    misinys <- cbind(data.frame(norm_duomenys$imtis), klase=misinys$klase)
```

```
    plot <- ggpairs(
      misinys, colour='klase', alpha=0.2,
      upper=list(continuous = 'points', combo = 'box'),
      lower=list(continuous = 'density', combo = 'facethist'),
      diag=list(continuous='density', discrete='bar'),
      axisLabels="show",
      title=bquote('Sferintas mišinys' ~ .(pav))
    )
    print(plot)
  }}
```

R failo vykd_r2_tankiai.r programos tekstas

```
source('funk_tp_algoritmas.r')
source('funk_gauso_misinys.r')
source('konf_misiniu_parametrai.r')
source('funk_paklaidos.r')
source('funk_2d_tankis.r')
require('ggplot2')
require("gridExtra")
# parametrai ------------------------------------------------------------------
turis <- 500
turis_tinkl <- 200
seed <- 1
n_krypciu <- 10
pol_eile <- 6
i_misinio <- 1
#sriftas <- 'sans'  # Arial
sriftas <- 'serif'  # Times
ar_brezti_kontura <- TRUE
ar_brezti_projekcija <- TRUE
ar_brezti_R <- TRUE
ar_brezti_tarpines_imtis <- TRUE
# -----------------------------------------------------------------------------
imtis <- generuoti_misini(turis, par_misiniu[[i_misinio]], seed, TRUE)
rez_imtis_norm <- normuoti_pradine_imti(imtis)
rez_tp <- atlikti_tikslini_projektavima(
  n_krypciu, rez_imtis_norm$imtis, pol_eile, TRUE)
dim <- ncol(imtis)
koef_denorm_tankis <- sqrt(prod(rez_imtis_norm$tikr_reiksmes))
tankio_ivertis_gauso <- dmvnorm(rez_imtis_norm$imtis, rep(0, dim), diag(dim)) /
  koef_denorm_tankis
xy <- gauti_xy_asis(data.frame(imtis), turis_tinkl)
tinklelis <- gauti_tinkleli(xy$x, xy$y)
tinklelis_norm <- normuoti_nauja_imti(
  tinklelis, rez_imtis_norm$vidurkiai,
  rez_imtis_norm$tikr_vektoriai,
  rez_imtis_norm$tikr_reiksmes)
tankis <- gauti_misinio_tanki(imtis, par_misiniu[[i_misinio]])
pakl1_gauso <- gauti_pakl1(tankis, tankio_ivertis_gauso)
pakl2_gauso <- gauti_pakl2(tankis, tankio_ivertis_gauso)
paklaidos <- data.frame(K=0, pakl1=1, pakl2=1)
df_imtis <- data.frame(imtis)
names(df_imtis) <- c('x1', 'x2')
ggplot(df_imtis, aes(x1, x2)) + geom_point() + coord_fixed() +
  ggtitle('Imtis') + theme_bw()
for (n_kr in 1:n_krypciu) {
  tankio_ivertis <- gauti_tankio_iverti(
    rez_imtis_norm$imtis,
    rez_tp$imtys, rez_tp$kryptys, n_kr, pol_eile) /
    koef_denorm_tankis
  paklaidos <- rbind(
    paklaidos,
    data.frame(
```

```
    K=n_kr,
    pakl1=gauti_pakl1(tankis, tankio_ivertis) / pakl1_gauso,
    pakl2=gauti_pakl2(tankis, tankio_ivertis) / pakl2_gauso))
  grafikai <- list()
if (ar_brezti_tarpines_imtis == TRUE) {
  df_tarpine_imtis <- data.frame(rez_tp$imtys[[n_kr]])
  names(df_tarpine_imtis) <- c('z1', 'z2')
  pav_sklaidos <- bquote(Z[.(n_kr - 1)] * ',' ~  alpha[.(n_kr)])
  if (n_kr > 1) {
    pav_sklaidos <- bquote(.(pav_sklaidos) ~ ir ~ alpha[.(n_kr - 1)])
  }
  grafikai[[1]] <- ggplot(df_tarpine_imtis, aes(z1, z2)) +
    labs(title=pav_sklaidos) +
    xlab(bquote(Z1)) +
    ylab(bquote(Z2)) +
    geom_point(alpha=0.3) +
    geom_abline(aes(
      intercept=0,
      slope=rez_tp$kryptys[[n_kr]][2] / rez_tp$kryptys[[n_kr]][1]),
      colour='red') +
    theme_bw() + coord_fixed()
  if (n_kr > 1) {
    grafikai[[1]] <- grafikai[[1]] +
      geom_abline(aes(
        intercept=0,
        slope=rez_tp$kryptys[[n_kr - 1]][2] / rez_tp$kryptys[[n_kr - 1]][1]),
        colour='red', linetype='longdash', alpha=0.7)
  } }
  if (ar_brezti_projekcija == TRUE) {
  #duom_proj <- tinklelis_norm %*% rez_tp$kryptys[[n_kr]]
  duom_proj_tarpines <- rez_tp$imtys[[n_kr]] %*% rez_tp$kryptys[[n_kr]]
  lp <- gauti_polinomus(2 * pnorm(duom_proj_tarpines) - 1, pol_eile)
  lp_koef <- gauti_polinomus(2 * pnorm(duom_proj_tarpines) - 1, pol_eile)
  koef <- colMeans(lp_koef) * seq(1, by=2, length=pol_eile + 1)
  coef_pr <- (lp %*% koef)
  tankis_pr <- coef_pr * dnorm(duom_proj_tarpines)
  tankis_pr[tankis_pr < 0] <- 0
  duom_norm_proj <- seq(from=min(duom_proj_tarpines),
                        to=max(duom_proj_tarpines), length.out=100)
  duom_norm_proj <- data.frame(cbind(duom_norm_proj, dnorm(duom_norm_proj)))
  names(duom_norm_proj) <- c('xn', 'pn')
  df_projekcija <- data.frame(cbind(duom_proj_tarpines, tankis_pr))
  names(df_projekcija) <- c('x', 'p')
  grafikai[[2]] <- ggplot(df_projekcija, aes(x, p)) +
    labs(title=bquote(
      widehat(p)[(.(n_kr))](X) * ',' ~
        X ~ '=' ~ alpha[(.(n_kr))]^'T' * Z[.(n_kr - 1)] * ',' ~
        widehat(I)(alpha[.(n_kr)]) ~ '=' ~
        .(sprintf('%.3f', rez_tp$indeksai[[n_kr]])))) +
    xlab(bquote(X)) +
    ylab(bquote(widehat(p))) +
    geom_line(alpha=0.8) +
    geom_line(aes(x=xn, y=pn), data=duom_norm_proj, colour='red',
              linetype='dotted', apha=0.8) +
    scale_y_continuous(limits=c(0, 0.75)) +
    theme_bw()
}
if (ar_brezti_kontura == TRUE) {
  tankio_ivertis_tinklelis <- gauti_tankio_iverti(
    tinklelis_norm,
    rez_tp$imtys, rez_tp$kryptys, n_kr, pol_eile) /
    koef_denorm_tankis
  tinklelis_3d <- data.frame(cbind(tinklelis, tankio_ivertis_tinklelis))
```

```
    names(tinklelis_3d) <- c('x1', 'x2', 'p')
    grafikai[[3]] <- ggplot(tinklelis_3d, aes(x1, x2, z=p)) +
      labs(title=bquote(widehat(p)[.(n_kr)](Z))) +
      xlab(bquote(Y1)) +
      ylab(bquote(Y2)) +
      #geom_tile(aes(fill=p)) +
      #stat_contour(geom="polygon", aes(fill=..level..)) +
      #stat_contour(size=2, color="gray") +
      stat_contour(size=0.6, aes(color=..level..)) +
      theme_bw() +
      coord_fixed()
  }
  if (ar_brezti_R == TRUE) {
    duom_proj_tarpines <- rez_tp$imtys[[n_kr]] %*% rez_tp$kryptys[[n_kr]]
    df_R <- data.frame(2 * pnorm(duom_proj_tarpines) - 1)
    names(df_R) <- c('r')
    grafikai[[4]] <- ggplot(df_R, aes(r)) +
      labs(title=bquote(
        R ~ '=' ~ 2 * Phi(X) - 1 )) +
      xlab(bquote(R)) +
      ylab(bquote(widehat(p))) +
      geom_density(alpha=0.8) +
      geom_hline(yintercept=0.5, colour='red', alpha=0.8,
                 linetype='dotted') +
      scale_y_continuous(limits=c(0, 1)) +
      theme_bw()
  }
  grafikai <- lapply(grafikai, function(g) {
    g + theme(plot.title=element_text(family=sriftas, size=12)) +
      theme(axis.title.x=element_text(family=sriftas, size=10)) +
      theme(axis.title.y=element_text(family=sriftas, size=10))
  })
  grafikas <- do.call(marrangeGrob, c(grafikai, list(
    nrow=2, ncol=2,
    top=textGrob(bquote(
      K ~ '=' ~ .(n_kr) * ',' ~
      alpha[.(n_kr)] ~ '=' ~
        (.(sprintf('%.3f', rez_tp$kryptys[[n_kr]][1])) * ', ' *
           .(sprintf('%.3f', rez_tp$kryptys[[n_kr]][2])))
    ), gp=gpar(fontfamily=sriftas,
               fontsize=12)))))
  ggsave(filename=paste0(
    'r2demo_mis', i_misinio, '_n', turis, '_k', n_kr, '_j', pol_eile, '.png'),
    path='grafikai/r2demo/',
    dpi=300,
    plot=grafikas)
}
```

R failo vykd_r2_tankiai.r programos tekstas

```
source('funk_tp_algoritmas.r')
source('funk_gauso_misinys.r')
source('konf_misiniu_parametrai.r')
source('funk_paklaidos.r')
source('funk_2d_tankis.r')
require('ggplot2')
require("gridExtra")
# parametrai ------------------------------------------------------------------
turis <- 500
turis_tinkl <- 200
seed <- 1
n_krypciu <- 10
pol_eile <- 6
i_misinio <- 1
```

```
#sriftas <- 'sans'  # Arial
sriftas <- 'serif'  # Times
ar_brezti_kontura <- TRUE
ar_brezti_projekcija <- TRUE
ar_brezti_R <- TRUE
ar_brezti_tarpines_imtis <- TRUE
# ------------------------------------------------------------------------
imtis <- generuoti_misini(turis, par_misiniu[[i_misinio]], seed, TRUE)
rez_imtis_norm <- normuoti_pradine_imti(imtis)
rez_tp <- atlikti_tikslini_projektavima(
  n_krypciu, rez_imtis_norm$imtis, pol_eile, TRUE)
dim <- ncol(imtis)
koef_denorm_tankis <- sqrt(prod(rez_imtis_norm$tikr_reiksmes))
tankio_ivertis_gauso <- dmvnorm(rez_imtis_norm$imtis, rep(0, dim), diag(dim)) /
  koef_denorm_tankis
xy <- gauti_xy_asis(data.frame(imtis), turis_tinkl)
tinklelis <- gauti_tinkleli(xy$x, xy$y)
tinklelis_norm <- normuoti_nauja_imti(
  tinklelis, rez_imtis_norm$vidurkiai,
  rez_imtis_norm$tikr_vektoriai,
  rez_imtis_norm$tikr_reiksmes)
tankis <- gauti_misinio_tanki(imtis, par_misiniu[[i_misinio]])
pakl1_gauso <- gauti_pakl1(tankis, tankio_ivertis_gauso)
pakl2_gauso <- gauti_pakl2(tankis, tankio_ivertis_gauso)
paklaidos <- data.frame(K=0, pakl1=1, pakl2=1)
df_imtis <- data.frame(imtis)
names(df_imtis) <- c('x1', 'x2')
ggplot(df_imtis, aes(x1, x2)) + geom_point() + coord_fixed() +
  ggtitle('Imtis') + theme_bw()
for (n_kr in 1:n_krypciu) {
  tankio_ivertis <- gauti_tankio_iverti(
    rez_imtis_norm$imtis,
    rez_tp$imtys, rez_tp$kryptys, n_kr, pol_eile) /
    koef_denorm_tankis
  paklaidos <- rbind(
    paklaidos,
    data.frame(
      K=n_kr,
      pakl1=gauti_pakl1(tankis, tankio_ivertis) / pakl1_gauso,
      pakl2=gauti_pakl2(tankis, tankio_ivertis) / pakl2_gauso))
    grafikai <- list()
  if (ar_brezti_tarpines_imtis == TRUE) {
    df_tarpine_imtis <- data.frame(rez_tp$imtys[[n_kr]])
    names(df_tarpine_imtis) <- c('z1', 'z2')
    pav_sklaidos <- bquote(Z[.(n_kr - 1)] * ',' ~  alpha[.(n_kr)])
    if (n_kr > 1) {
      pav_sklaidos <- bquote(.(pav_sklaidos) ~ ir ~ alpha[.(n_kr - 1)])
    }
    grafikai[[1]] <- ggplot(df_tarpine_imtis, aes(z1, z2)) +
      labs(title=pav_sklaidos) +
      xlab(bquote(Z1)) +
      ylab(bquote(Z2)) +
      geom_point(alpha=0.3) +
      geom_abline(aes(
        intercept=0,
        slope=rez_tp$kryptys[[n_kr]][2] / rez_tp$kryptys[[n_kr]][1]),
        colour='red') +
      theme_bw() + coord_fixed()
    if (n_kr > 1) {
      grafikai[[1]] <- grafikai[[1]] +
        geom_abline(aes(
          intercept=0,
          slope=rez_tp$kryptys[[n_kr - 1]][2] / rez_tp$kryptys[[n_kr - 1]][1]),
```

```
        colour='red', linetype='longdash', alpha=0.7)
  }  }
  if (ar_brezti_projekcija == TRUE) {
  #duom_proj <- tinklelis_norm %*% rez_tp$kryptys[[n_kr]]
  duom_proj_tarpines <- rez_tp$imtys[[n_kr]] %*% rez_tp$kryptys[[n_kr]]
  lp <- gauti_polinomus(2 * pnorm(duom_proj_tarpines) - 1, pol_eile)
  lp_koef <- gauti_polinomus(2 * pnorm(duom_proj_tarpines) - 1, pol_eile)
  koef <- colMeans(lp_koef) * seq(1, by=2, length=pol_eile + 1)
  coef_pr <- (lp %*% koef)
  tankis_pr <- coef_pr * dnorm(duom_proj_tarpines)
  tankis_pr[tankis_pr < 0] <- 0
  duom_norm_proj <- seq(from=min(duom_proj_tarpines),
                        to=max(duom_proj_tarpines), length.out=100)
  duom_norm_proj <- data.frame(cbind(duom_norm_proj, dnorm(duom_norm_proj)))
  names(duom_norm_proj) <- c('xn', 'pn')
  df_projekcija <- data.frame(cbind(duom_proj_tarpines, tankis_pr))
  names(df_projekcija) <- c('x', 'p')
  grafikai[[2]] <- ggplot(df_projekcija, aes(x, p)) +
    labs(title=bquote(
      widehat(p)[(.(n_kr))](X) * ',' ~
        X ~ '=' ~ alpha[(.(n_kr))]^'T' * Z[.(n_kr - 1)] * ',' ~
        widehat(I)(alpha[.(n_kr)]) ~ '=' ~
        .(sprintf('%.3f', rez_tp$indeksai[[n_kr]])))) +
    xlab(bquote(X)) +
    ylab(bquote(widehat(p))) +
    geom_line(alpha=0.8) +
    geom_line(aes(x=xn, y=pn), data=duom_norm_proj, colour='red',
              linetype='dotted', apha=0.8) +
    scale_y_continuous(limits=c(0, 0.75)) +
    theme_bw()
}
if (ar_brezti_kontura == TRUE) {
  tankio_ivertis_tinklelis <- gauti_tankio_iverti(
    tinklelis_norm,
    rez_tp$imtys, rez_tp$kryptys, n_kr, pol_eile) /
    koef_denorm_tankis
  tinklelis_3d <- data.frame(cbind(tinklelis, tankio_ivertis_tinklelis))
  names(tinklelis_3d) <- c('x1', 'x2', 'p')
  grafikai[[3]] <- ggplot(tinklelis_3d, aes(x1, x2, z=p)) +
    labs(title=bquote(widehat(p)[.(n_kr)](Z))) +
    xlab(bquote(Y1)) +
    ylab(bquote(Y2)) +
    #geom_tile(aes(fill=p)) +
    #stat_contour(geom="polygon", aes(fill=..level..)) +
    #stat_contour(size=2, color="gray") +
    stat_contour(size=0.6, aes(color=..level..)) +
    theme_bw() +
    coord_fixed()
}
if (ar_brezti_R == TRUE) {
  duom_proj_tarpines <- rez_tp$imtys[[n_kr]] %*% rez_tp$kryptys[[n_kr]]
  df_R <- data.frame(2 * pnorm(duom_proj_tarpines) - 1)
  names(df_R) <- c('r')
  grafikai[[4]] <- ggplot(df_R, aes(r)) +
    labs(title=bquote(
      R ~ '=' ~ 2 * Phi(X) - 1 )) +
    xlab(bquote(R)) +
    ylab(bquote(widehat(p))) +
    geom_density(alpha=0.8) +
    geom_hline(yintercept=0.5, colour='red', alpha=0.8,
               linetype='dotted') +
    scale_y_continuous(limits=c(0, 1)) +
    theme_bw()
```

```
  }
  grafikai <- lapply(grafikai, function(g) {
    g + theme(plot.title=element_text(family=sriftas, size=12)) +
      theme(axis.title.x=element_text(family=sriftas, size=10)) +
      theme(axis.title.y=element_text(family=sriftas, size=10))
  })
  grafikas <- do.call(marrangeGrob, c(grafikai, list(
    nrow=2, ncol=2,
    top=textGrob(bquote(
      K ~ '=' ~ .(n_kr) * ',' ~
      alpha[.(n_kr)] ~ '=' ~
        (.(sprintf('%.3f', rez_tp$kryptys[[n_kr]][1])) * ', ' *
           .(sprintf('%.3f', rez_tp$kryptys[[n_kr]][2])))
    ), gp=gpar(fontfamily=sriftas,
               fontsize=12)))))
  ggsave(filename=paste0(
    'r2demo_mis', i_misinio, '_n', turis, '_k', n_kr, '_j', pol_eile, '.png'),
    path='grafikai/r2demo/',
    dpi=300,
    plot=grafikas) }
```

R failo vykd_i_2p_grafikai.r programos tekstas

```
require("ggplot2")
require("gridExtra")
source("funk_grafiku_antrastes.r")
source("konf_tyrimo_parametrai.r")
source.with.encoding('konf_misiniu_parametrai.r', encoding='utf-8')
source('funk_grafikai.r')
load(file="tp_paklaidos_.RData")
### rezultatu filtravimo parametrai; NULL - nefiltruoti
filt_misiniai <- NULL
# filt_misiniai <- c('12_R5-lbp')
filt_turiai <- NULL
# filt_turiai <- c('1000')
filt_eiles <- NULL
# filt_eiles <- c(6, 8)
### Grafiku braizymo parametrai
n_krypciu_grafikui <- 100
ar_braizyti_pi <- TRUE
slenkstis_1 <- 0.15
slenkstis_2 <- 0.15
f_misiniai <- sukurti_filtra(filt_misiniai, names(rez_graf))
for (i_misinio in f_misiniai()) {
  dim <- length(par_misiniu[[i_misinio]][[1]][[2]])
  f_turiai <- sukurti_filtra(filt_turiai, names(rez_graf[[i_misinio]]))
  for (tur in f_turiai()) {
    pav <- bquote(
      Y %in% R^.(dim) * ',' ~
        .(persidengimo_pav[[strsplit(i_misinio, '-')[[1]][2]]]) * ',' ~
        tiksliniam ~ projektavimui ~ N ~ '=' ~ .(tur))
    grafikai1 <- list()
    grafikai1[[1]] <- graf_indeksai(
      gauti_duomenis_grafikui(
        rez_graf[[i_misinio]][[tur]]$df_indeksai,
        n_krypciu_grafikui, filt_eiles),
      ar_braizyti_pi, slenkstis_1, slenkstis_2)
    for (i_pakl in 1:2) {
      grafikai1[[i_pakl + 1]] <- graf_paklaidos(
        gauti_duomenis_grafikui(
          rez_graf[[i_misinio]][[tur]][[paste0('df_paklaidos',
                                               i_pakl, 'norm')]],
          n_krypciu_grafikui, filt_eiles),
        i_pakl, ar_braizyti_pi)
```

```
    }
    grafikas1 <- do.call(marrangeGrob, c(grafikai1, list(
      nrow=1, ncol=3,
      top=textGrob(pav, gp=gpar(fontfamily=sriftas, fontsize=12)))))
    failo_pav <- paste0(
      'mis', i_misinio, "_n", tur, '_im', n_imciu,
      ifelse(! is.null(filt_eiles),
             paste0('_j', paste0(filt_eiles, collapse='-')), ''),
      ifelse(ar_braizyti_pi, '_pi', ''))
    ggsave(filename=paste0(
      failo_pav, '_ind_2pakl.png'),
      path='i_2p_grafikai/',
      dpi=100,
      plot=grafikas1)  }}
```

R failo exp_visi_duomenys.r programos tekstas

```
require('reshape2')
source.with.encoding('konf_misiniu_parametrai.r', encoding='utf-8')
failo_pav <- 'csv/visi_duomenys.csv'
NA_isvedimui <- 'NA'
load(file='tp_paklaidos_.RData')
df_rysiai <- list(
  df_indeksai='indeksas',
  df_paklaidos1norm='sigma_1',
  df_paklaidos2norm='sigma_2',
  df_mardia_skew='mardia_skew',
  df_mardia_kurt='mardia_kurt',
  df_hz='hz',
  df_royston='royston'
)
gauti_misinio_eilutes <- function(df_list_rez_graf)
{
  df_ilga <- do.call(
    rbind,
    lapply(
      names(df_rysiai),
      function(df_pav, df_list, df_rysiai) {
        return(cbind(list(laukas=df_rysiai[[df_pav]]),
                     df_list[[df_pav]][, c('K', 'J', 'vid_reiksme')]))
      },
      df_list_rez_graf, df_rysiai))

  df_plati <- dcast(df_ilga, J ~ laukas + K, value.var='vid_reiksme')
    return(df_plati)
}
df_eksportavimui <- data.frame()
for (i_misinio in names(rez_graf)) {
  for (turis in names(rez_graf[[i_misinio]])) {
    print(i_misinio)
    print(turis)
    df_eksportavimui <- rbind(
      df_eksportavimui,
      cbind(
        list(
          pav=i_misinio,
          dim=length(par_misiniu[[i_misinio]][[1]][[2]]),
          tipas=persidengimo_pav[[strsplit(i_misinio, '-')[[1]][2]]]),
        turis=turis,
        gauti_misinio_eilutes(rez_graf[[i_misinio]][[turis]])))
  }}
write.csv2(df_eksportavimui, failo_pav, row.names=FALSE, na=NA_isvedimui)
```