

Bayesi statistika kasutades R keelt

*Taavi Päll
Ülo Maiväli*

Contents

Saateks	5
I OSA	7
1 Sissejuhatus: maailm, teooria ja mudel	9
Suur ja väike maailm	9
Mudeli väike maailm	12
2 Küsimused, mida statistika küsib	17
Jäta meelde	18
3 Kuidas näevad välja teie andmed	19
Summaarsed statistikud	19
Keskvärtused	19
Muutuja sisene varieeruvus	20
Logaritmi andmed	21
Iseloomustada andmeid algses skaalas: mediaan (MAD)	25
Muutujate koosvarieeruvus	26
4 Lineaarsed mudelid	31
4.1 Sirge võrrand	31
4.2 Ennustus lineaarsest mudelist	36
4.3 Neli mõistet	37
4.4 Mudeli fittimine	38
4.5 Lineaarse regressiooni eeldused	41
5 Kaks lineaarse mudeli laiendust	45
5.1 Mitme sõltumatu prediktoriga mudel	45
5.2 Interaktsionimudel	48
6 Vähimruutude meetodiga fititud mudelite töövoog – lm()	55
1. vaatame mudeli koefitsiente	55
2. Testime mudeli eeldusi	56

Residuaalid y ja x muutujate vastu	63
3. Teeme mudeli põhjal ennustusi (marginal plots)	65
4. Võrdleme mudeleid	67
7 Andmete transformeerimine	69
7.1 Logaritmimine	69
7.2 Standardiseerimine	72
7.3 Tsentreerimine	73
7.4 Mudeli koefitsientide transformeerimine	73
7.5 Pidev või diskreetne muutuja?	73
7.6 mitmese regressiooni üldised printsibid	74
8 Veamudel	75
8.1 Lihtne varieeruvuse mudel	75
8.2 protsessimudel ja varieeruvuse mudel lineaarses regressioonis	78
Enimkasutatud veamudel on normaaljaotus	79
Normaaljaotuse ja lognormaaljaotuse erilisus	87
8.3 Teised veamudelid	89
9 EDA — eksploratoorne andmeanalüüs	93
9.1 EDA kokkuvõte	97
10 Lausearvutuslik loogika	99
10.1 Tõetabel	100
10.2 Konjunksioon	100
10.3 Disjunksioon	101
10.4 Konditsionaal	103
10.5 Tautoloogia ja kontradiktsioon	104
10.6 loogiline argument ja valiidne järeldamine	104
10.7 Modus Ponens ja Modus Tollens	106
10.8 Lausearvutusest tõenäosuste loogikasse	108
11 Järel dav statistika	111
Järel dav statistika on tõenäosusteooria käepikendus	112
Andmed ei ole sama, mis tegelikkus	122
II OSA	127
12 Bootstrap	129
12.1 Mõned tava-bootstrapi paketid	133
Bayesi bootstrap	134
Parameetriline bootstrap	135
Bootstrappimine ei ole kogu tõde	138
13 Bayesi põhimõte	141
Esimene näide	142

CONTENTS	5
Teine näide: sõnastame oma probleemi ümber	147
Kui $n = 1$	149
14 Mudelite keel	157
Beta prior	160
Prioritest üldiselt	161
15 MCMC põhimõte	165
16 Lihtne normaaljaotuse mudel	167
Kui lai on meie töepärafunktsioon?	170
Lihtne või robustne normaalne mudel?	170
MCMC ahelete kvaliteet	173
Lineaarne regressioon	184
<code>lm()</code> - vähimruutude meetodiga fititud lineaarsed mudelid	184
17 Bayesi meetodil lineaarse mudeli fittimine	191
Ennustused mudelist	196
Lognormaalne töepäramudel	199
18 Mitme prediktoriga lineaarne regressioon	205
Miks multivariaatsed mudelid head on?	208
Mudeldamine standardiseeritud andmetega	209
18.1 Prediktorite valik e milline on parim mudel	210
19 Keerulisemate mudelitega töötamine	217
Predictor residual plots	217
Ennustavad plotid	219
Posterior prediction plots	221
Interaktsioonid prediktorite vahel	225
Interaktsioonid pidevatele tunnustele	230
20 Mitmetasemelised mudelid	237
20.1 kahetasemeline mudel algebra keeles	238
20.2 mitmetasemeline mudel R-i mudelikeeles	242
20.3 Mitmetasemeliste mudelite lisaeeldused	243
20.4 Mitmetasemeline mudel töötab korraga mitmel tasmel	243
20.5 ANOVA-laadne mudel	247
20.6 Vabad interceptid klassikalises regressioonimudelis	250
20.7 Vabad tõusud ja interceptid	256
20.8 Hierarhiline mudel pidevate prediktoritega	262
21 brms	267
21.1 brms-i töövoog	268
21.2 Mudeli eelduste kontroll	301
22 Brms mudelid	309

22.1 Robustne lineaarne regressioon	309
22.2 lognormaalne töepärafunktsioon	317
22.3 Puuduvate andmete imputatsioon	321
22.4 Binoomjaotusega mudelid	323
22.5 Monotoonilised efektid	337
23 brms süntaks	341
23.1 General formula structure	342
23.2 brms likelihoods	348
23.3 priors	350
24 Puuduvad andmed	355
24.1 Andmete puudumise mehhanismid:	356
24.2 Ühekaupa imputatsiooni meetodid	357
24.3 Mitmene imputatsioon	359
24.4 Predictive mean matching	365
24.5 Imputeerimine mitte-normaalsete jaotuste korral	365
24.6 Kategoorilised muutujad	365
24.7 countid (sisaldavad nulle)	366
24.8 Graafilised meetodid	366
24.9 Tulemuste avaldamine	371
A Bayesi ja sagedusliku statistika võrdlus	373
Kaks statistikat: ajaloost ja tõenäosusest	373
Poleemika: kumbki tõenäosus pole päris see, mida üldiselt arvatakse .	375
Võrdlev näide: kahe grupi võrdlus	376
Kahe paradigma erinevused	382
Statistiline ennustus kui mitmetasandiline protsess	384
B Sõnastik	389
Kaks statistikat: ajaloost ja tõenäosusest	405
Poleemika: kumbki tõenäosus pole päris see, mida üldiselt arvatakse .	407
Võrdlev näide: kahe grupi võrdlus	408
Kahe paradigma erinevused	414
Statistiline ennustus kui mitmetasandiline protsess	416

Saateks

See õpik soovib anda praktilisi andmeanalüüs oskusi töötamiseks reaalsete andmetega. See puudutab laias laastus kolme teemat:

1. Kuidas summeerida andmeid: keskmise, varieeruvuse ja kovarieeruvuse näitajad.
2. Kuidas graafiliste meetodite abil kontrollida andmete kvaliteeti ja püstitada uusi hüpoteesi.
3. Kuidas teha andmete põhjal järeldusi protsessi kohta, mis neid andmeid genereerib, ühtlasi kirjeldades adekvaatselt neid järeldusi ümbritsevat ebakindlust.

Kuna me püüame anda eeskätt praktilisi oskusi andmeanalüüsiks, mitte matemaatilist ega muidu teoreetilist haridust, siis keskendume moodsatel bayesi meetoditele. Need on küll arvuti jaoks töömahukamat kui klassikalised statistilisel olulisel põhinevad testid, aga inimese jaoks kergemini õpitavad ning tõlgendataavad. Klassikalist ja bayesi statistikat võrdleme lisas 1. Bayesiaanliku lähenemise hind on, et kasutaja peab omandama vähemalt algtasemel R keele, mis võimaldab käsurealt anmdeid manipuleerida. R-i õppimine nõub kahtlemata lisapingutust, aga me usume, et see tasub ära igaühele, kes töötab vähemalt keskmise suurusega andmekogudega. Me teeme sissejuhatuse R-i peatükkides . . ., keskendudes R-i tidyverse ökosüsteemile, mis on optimeeritud olema kergesti kasutatav ja õpitav inimestele, kelle põhitöö ei ole seotud koodi kirjutamisega.

Pingutus R õppimiseks tasub teile mitmel erineval viisil. R võimaldab palju kiiremini andmetabeleid analüüsiks sobivasse vormi ajada kui spreadsheet programmid. R-i graafikasüsteem, eriti ggplot2, on võimas ning paindlik tööriist väga erinevate graafikute koostamiseks. R-s jooksevad praktiliselt kõik statistilised testid, mida inimmõistus on loonud – R on levinuim statistikaprogramm maailmas, mis on eriti hästi sobiv statistiliseks modelleerimiseks. See tähendab ka, et üle maailma on suur seltskond inimesi, kes R-i arendab ja on nõus vastama ka teie küsimustele. Lisaboonusena on juba kord salvestatud R-i koodi taaskasutades palju lihtsam oma analüüs korrrata ja vastavalt vajadustele muuta, kui spreadsheetide puhul.

Õpiku kasutamise eeldused: - Arvuti. - Matemaatikaoskused, mis hõlmavad liitmist, lahutamist, korrutamist, jagamist, logaritmimist ja astendamist. - Kõrgemat matemaatikat me ei vaja.

Part I

OSA

Chapter 1

Sissejuhatus: maailm, teooria ja mudel

Suur ja väike maailm

Kuna maailmas on kõik kõigega seotud, on seda raske otse uurida. Teadus töötab tänu sellele, et teadlased lõikavad reaalsuse väikesteks tükkideks, kasutades tordilabidana teaduslike hüpoteese, ning uurivad seda tüki kaupa lootuses, et kui kõik tükid on korralikult läbi nätsutatud, saab sellest taas tordi kokku panna. Tüüpiline bioloogiline hüpotees pakub välja tavakeelse (mitte matemaatilise) seletuse mõnele piiritletud loodusnähtusele.

Näiteks antibiootikume uuritakse keemilise sideme tasemel kasutades orgaanilise keemia meetodeid. Antibiootikumide molekulaarseid märklaudu uuritakse molekulaarbioloogiliste meetoditega, nende toimet uuritakse rakubioloogia ja füsioloogia meetoditega, aga kaasajal on väga olulised ka ökoloogilised, evolutsionilised, meditsiinilised, põllumajanduslikud, majanduslikud ja psühholoogilised aspektid. Kõigil neil tasanditel on loodud palju hüpoteese, millega kokku moodustub meie teadmine antibiootikumide kohta. Neid väga erinevaid asju, mida me kutsume hüpoteesideks saab sageli jagada osadeks (ja neid osa-hüpoteese omakorda osadeks), mida saab omakorda osaliselt kirjeldada matemaatiliste formalismide ehk mudelite abil. Ja neid mudeliteid saab võrrelda andmetega. Kuigi erinevate tasemetega hüpoteesid on tavakeeles üksteisest väga erinevad, on neid kirjeldavad mudelid sageli matemaatiliselt sarnased.

Kui mudel on teooria lihtsustus, siis teooria on maailma lihtsustus.

Mudeliteks nimetatakse bioloogias väga erinevaid asju: skeeme, diagramme, füüsikalisi mudeliteid (näit Watsoni ja Cricki poolt kasutatud nukleotiidi mudeliteid), mudelorganisme, katsesüsteeme, matemaatilisi mudeliteid jms. Üldiselt teeb

mudeli mudeliks, et see asendab selle, mida teadlane tegelikult uurida tahab millegagi, mida on lihtsam mõista, manipuleerida või uurida. Meie räägime edaspidi ainult matemaatilisest mudelist ja eriti selle erijuhist, statistilisest ehk stohhastilisest mudelist.

Mis juhtub, kui teie mudel, ja seega ka hüpotees, mis selle mudeli genereeris, on andmetega kooskõlas? Kas see tähendab, et see hüpotees vastab töele? Või, et see on töenäoliselt tõene? Kahjuks on vastus mõlemale küsimusele eitav. Põjhuseks on asjaolu, et enamasti leiab iga nähtuse seletamiseks rohkem kui ühe alternatiivse teadusliku hüpoteesi ning rohkem kui üks üksteist välistav hüpotees võib olla olemasolevate andmetega võrdses kooskõlas. Asja teeb veelgi hullemaks, et teoreetiliselt on võimalik sõnastada lõpmata palju erinevaid teooriaid, mis kõik pakuvad alternatiivseid ja üksteist välistavaid seletusi samale nähtusele. Kuna hüpoteeesse on lõpmatu hulk, aga andmete hulk on alalti lõplik, siis saab igas teaduslikus faktis kahelda.

Ei saa kindel olla, et parimad teooriad on meile üldse kunagi pähe torganud ning, et meie poolt kogutud vähesed andmed kajastavad hästi tegelikkust.

Ca. 1910 mõtlesid Bertrand Russell ja G.E. Moore välja töe vastavusteooria, mille kohaselt töest lausungit eristab väärast vastavus füüsikalisele maailmale. Seega on töesed ainult need laused, mis vastavad asjadele. Ehkki keegi ei oska siiani öelda, mida vastavus selles kontekstis tähendab või kuidas seda saavutada, on vastavusteoria senini köige populaarsem töeteooria filosoofide hulgas (mis on kõnekas alternatiivide kohta). Samamoodi, kui lausete vastavusest maailmaga, võime rääkida ka võrrandite (ehk mudelite) vastavusest lausetega. Vastavusest lausetaga sellepärast, et mudelid on loodud kirjeldama teaduslikke teooriaid, mitte otse maailma. Seega ei pea me muretsema mudelite töeväärtuse pärast. Võib isegi väita, et mudeli töeväärtusest rääkimine on kohatu.

(1) Näide: politoloogia.

Meil on hüpotees (H1), mille kohaselt demokraatlikus süsteemis käituvald valijad ratsionaalselt ehk lähtuvalt endi huvidest (Achen and Bartels 2016). Alternatiiv (H2) ütleb, et valijad ei vali poliitikuid lähtuvalt oma tegelikest huvidest. Kuna H1 on liiga lai, et seda otse andmetega võrrelda, tuletame sellest kitsama alamhüpoteesi (H1.1), mille kohaselt valijad eelistavad tagasi valida kandidaate, kes on ennast töestanud sellega, et saavad hakkama majanduse edendamisega. Seega, poliitikud, kes on võimekad majanduse vallas, valitakse tagasi suurema töenäosusega kui need, kes seda ei ole. Sellest hüpoteesist tuletati kaks andemetest vastu testitavat järelmit: - H1.1.1 – majandusel läheb keskeltläbi paremini juba tagasi valitud poliitikute all kui esimest korda valitud poliitkute all, kelle ridu ei ole veel elektoraadi poolt harvendatud ja - H1.1.2 – majandusnäitajate varieeruvus on esimesel juhul väiksem, sest kehvemad poliitikud on juba valimist eemaldatud. Esimese järelmi testimiseks kasutati statistilise mudelina (m1) aritmeetilist keskmist koos standardveaga ja teise järelmi jaoks (m2) standardhälvet.

Tulemused olid paraku vastupidised H1.1.1 ja H1.1.2 poolt ennustatuga, millest autorid tegid järelduse, et olemasolevad andmed ei toeta hüpoteesi H1.1 (andmete vähesuse tõttu nad ei arvanud, et nad oleksid H1.1-e ümber lükanud). Seega, andmed fititi mudelitesse m1 ja m2, nende fittide põhjal tehti järeldused H1.1 ja H1.1.2 kohta (et m1 ja H1.1.1 ning m2 ja H1.1.2 vahel puudub kooskõla), mille põhjal omakorda tehti järeldus H1.1 kohta (et H1.1-e ei õnnestunud kinnitada), mille põhjal ükski ei tehtud formaalset järeldust H1 kohta. H1 vs. H2 kohta tehakse järeldus alles raamatut lõpus, lähtudes H1.1, H1.2, ..., H1.n kohta tehtud järeldustest.

(2) Näide: populatsioonigeneetika.

Populatsioonigeneetikas on evolutsioon defineeritud kui alleelide sageduste muutumine põlvkonnast põlvkonda. Kõigepealt defineeriti tingimused, milliste kehtimisel alleelide sagedus EI muutu. Need on juhuslik sigimine populatsioonis, lõpmata suur populatsioon, mis koosneb diploidsetest organismidest, kellel on 1 geneetiline lookus ja 2 alleeli. See on Hardy-Weinbergi printsip, millel põhineb enamus klassikalisest populatsioonigeneetikast ja mida kirjeldab võrrand

$$p^2 + 2pq + q^2 = 1$$

kus p^2 , $2pq$ ja q^2 on genotüüpide AA , Aa ja aa sagedused sugurakkudes ning p ja q on alleelide A ja a sagedused (ning $p + q = 1$). Populatsioonis, mis on Hardy-Weinbergi tasakaalus, on p ja q põlvkondade välitel muutumatud. Selleks, et tasakaalu lõhkuda, toome mudelisse lisaparametri w , mis iseloomustab valikusurvet ehk kohasust (fitnessi). Kohasus iseloomustab looduliku valiku poolt tingitud genotüüpide sageduste muutust populatsioonis. Nüüd saame deterministliku mudeli (deterministliku, sest mudeli parameetritele kindlad väärtsed omistades ja mudeli läbi arvutades saame vastuseks sama arvu, ükskõik mitu korda me seda arvutust ka ei kordaks):

$$p^2wAA + 2pqwAa + q^2waa = w_{mean}$$

kus w_{mean} on populatsiooni keskmise kohasus, wAA on genotüübi AA kohasus jne. Kui me teame parameetrite p, q, wAA, wAa ja waa väärtsusi, saame hõlpsasti arvutada populatsiooni kohasuse.

Vaadates maailma mudeli pilgu läbi, juhul kui looduslike mõõdetud genotüüpide sageduse muutus erineb mudelist arvutatud w_{mean} -ist, siis on meil tegemist geneetilise triiviga. Geneetiline triiv on genotüübisaageduste juhuslik muutus populatsioonis, mis on seda suurem, mida väiksem on populatsioon ja mida väiksem on valikusurve populatsioonile. Seega oleks nagu võimalik geneetilise triivi olemasolu tuvastada alati, kui empiiriline genotüübisaageduste muutuse kiirus erineb mudeli punktenustusest w_{mean} . Selle deterministliku mudeli järgi on valik ja triiv teineteist välistavad: kui empiiriline kohasus = w_{mean} , siis valik; muidu triiv.

Samas, kui me eeldame, et populatsiooni suurus ei ole lõpmata suur, tuleb mudelisse sisse juhuslik valimiviga. Mida väiksem on populatsioon, seda suurema tõenäosusega ei anna juhuslik paljunemine ka ilma valikusurveta populatsioonis järgmist põlvkonda, mille genotüübisedustele (ptk xxx simuleerime me juhuslikku valimiviga normaaljao-tuse mudelist). Seega muutub meie deterministlik mudel stohhastiliseks mudeliks, mille väljund ei ole enam punktväärtus w_{mean} -le vaid rida tõenäosusi erinevatele w_{mean} -i väärustustele (**sellise mudeli kuju vt ptk xxx**). Selle mudeli järgi ei ole valik ja triiv enam erinevat tüüpi protsessid, vaid ühe kontiinumi kaks poolust; kontiinumi, mis sõltub populatsiooni suurusest ja valikusurve tugevusest. Kuna puhas looduslik valik saab mudeli järgi toimuda ainult lõpmata suures populatsionis, milliseid looduses ei leidu, siis on alleeli a sageduse muutus teadlase poolt uuritavas looduslikus populatsionis x ühtaegu nii loodusliku valiku kui geenitriivi tagajärg.

Mis juhtub, kui me ei tee mudeli struktuurist otse järeldusi maailma kohta? Nüüd alustame me eeldusest, et looduslik valik on looduses reaalselt toimuv protsess. Näiteks Darwin nägi valikut loodusliku põhjusliku protsessina, mis on samas stohhastiline (mitte kõik kõrgema kohasusega organismid ei anna järglasi). Selle vaate kohaselt on loodusliku valiku tagajärjeks kallutatud valim genotüüpidest, mille avaldumise poolt põhjustatud erinevused organismides viisid nende erinevale paljunemisedukusele. Seega on valik ja triiv erinevat tüüpi looduslikud protsessid, mitte mudeli väljundid. Niisiis teeme rangelt vahet valikul ja triivil nende põhjuste järgi. Kui tõuseb kasulike genotüüpidega organismide osakaal, siis on tegemist loodusliku valiku poolt tingitud evolutsiooniga. Kui aga genotüüpide sageduste muutumine ei ole põhjustatud indiviidide füüsilikatest erinevustest, siis on tegu geneetilise triivi poolt tingitud evolutsiooniga.

Nõnda saame evolutsioniteooriast lähtudes hoopis teistsuguse vaate bioloogiale, kui mudeliteid otse tõlgendades. Muidugi ei tähenda see, et me ei vaja mudeliteid. Vajame küll, aga me peame neid ettevaatlikult tõlgendama, pidades silmas oma teooriate sisu. Andemetega fititud mudeliteid tõlgendamene ja seda ei tohiks kunagi teha otse mudelist päris maailmale.

Mudeli väike maailm

Ülalmainitud teadusliku meetodi puudused tingivad, et meie huvides on oma teaduslike probleeme veel ühe taseme võrra lihtsustada, taandades need statistilisteks probleemideks. Selleks tuletame tavakeelsest teaduslikust teooriast täpselt formuleeritud matemaatilise mudeli ning seejärel asume uurima oma mudelite lootuses, et mudeli kooskõla andmetega ütleb meile midagi teadusliku hüpoteesi kohta. Enamasti töötab selline lähenemine siis, kui mudeli ehitasisel arvestati võimaliku andmeid genereeriva mehanismiga – ehk, kui mudeli matemaatiline struktuur koostati teaduslikku hüpoteesi silmas pidades. Mudelid, mis ehatakse silmas pidades puhtalt matemaatilist sobivust andmetega, ei kipu

omama teaduslikku seletusjõudu, kuigi neil võib olla väga hea ennustusjõud.

Meil on kaks hüpoteesi, A ja B. Juhul kui A on tõene ja B on väär, kas on võimalik, et B on töele lähemal kui A? Kui A ja B on teineteist välistavad punkthüpoteesid parameetri värtuse kohta, siis on vastus eitav. Aga mis juhtub, kui A ja B on statistilised mudelid? Näiteks, kui tõde on, et eesti meeste keskmise pikkuse on 178.3 cm ja A ütleb, et keskmise pikkuse jäääb kuhugi 150 cm ja 220 cm vaheline ning B ütleb, et see jäääb kuhugi 179 cm ja 182 cm vaheline, siis on B "töele lähemal" selles mõttes, et meil on temast teaduslikus mõttes rohkem kasu. Siit on näha oluline erinevus teadusliku hüpoteesi ja statistilise mudeli vahel: hüpotees on orienteeritud töele, samal ajal kui mudel on orienteeritud kasule.

Mudeli maailm erineb päris maailmast selle poolest, et mudeli maailmas on kõik sündmused, mis põhimõtteliselt võivad juhtuda, juba ette teada ja üles loendatud (seda sündmuste kogu kutsutakse parameetriruumiks). Tehniliselt on mudeli maailmas üllatused võimatum.

Lisaks, tõenäosusteooriat, ja eriti Bayesi teoreemi, kasutades on meil garantii, et me suudame mudelis leiduva informatsiooniga ümber käia parimal võimalikul viisil. Kõik see rõõm jäääb siiski mudeli piiridesse. Mudeli eeliseks teoria ees on, et hästi konstrueeritud mudel on lihtsamini mõistetav — erinevalt vähegi keerulisemast teaduslikust hüpoteesist on mudeli eeldused ja ennustused läbinähavad ja täpselt formuleeritavad. Mudeli puuduseks on aga, et erinevalt teoriast ei ole mingit võimalust, et mudel vastaks tegelikkusele. Seda sellepärast, et mudel on taotluslikult lihtsustav (erandiks on puhtalt ennustuslikud mudelid, mis on aga enamasti läbinähtamatu struktuuriga). Mudel on kas kasulik või kasutu; teoria on kas tõene või väär. Mudeli ja maailma vahel võib olla kaudne peegeldus, aga mitte kunagi otsene side. Seega, ükski number, mis arvutatakse mudeli raames, ei kandu sama numbrina üle teaduslikku ega päris maailma. Ja kogu statistika (ka mitteparameetriseline) toimub mudeli väikses maailmas. Arvud, mida statistika teile pakub, elavad mudeli maailmas; samas kui teie teaduslik huvi on suunatud päris maailmale. Näiteks 95% usaldusintervall ei tähenda, et te peaksite olema 95% kindel, et tõde asub selles intervallis – sageli ei tohiks te seda nii julgelt tõlgendada isegi kitsas mudeli maailmas.

(3) Näide: Aristoteles, Ptolemaios ja Kopernikus

Aristoteles (384–322 BC) lõi teooria maailma toimimise kohta, mis domineeris haritud eurooplase maailmapilti enam kui 1200 aasta vältel. Tema ühendteooria põhines maailmapildil, mis oli üldtunnustatud juba sajandeid enne Aristotelest ja järgneva 1500 aasta jooksul kahtlesid selles vähesed mõistlikud inimesed. Selle kohaselt asub universumi keskpunktis statsionaarne maakera ning kõik, mida siin leida võib, on tehtud neljast elemendist: maa, vesi, õhk ja tuli. Samas, kogu maailmaruum alates kuu sfäärist on tehtud viiendast elemendist (eeter), mida aga ei leidu maal (nagu nelja elementi ei leidu kuu peal ja sealt edasi). Taevakehad (kuu, päike, planeedid ja kinnistähed) tiirlevad ümber maa kontsentrilistest

Schema huius praenitie divisionis Sphaerarum.



Figure 1.1: Keskaegne aristotellik maailm.

sfäärides, mille vahel pole vaba ruumi. Seega on kogu liikumine eetri sfäärides ühtlane ja ringikujuline ja see liikumine põhjustab pika põhjus-tagajärg ahela kaudu kõiki liikumisi, mida maapeal kohtame. Kaasa arvatud sündimine, elukäik ja surm. Kõik, mis maapeal huvitavat, ehk kogu liikumine, on algsest põhjustatud esimese liikumise poolt, mille käivitat kõige välimises sfääris paiknev meie jaoks mõistetamatu intellektiga "olend".

Aristotelese suur teooria ühendab kogu maaailmapildi alates meie mõistes keemiatest ja kosmoloogiast kuni bioloogia, maateaduse ja isegi geograafiani. Sellist ühendteooriat on erakordselt raske ümber lükata, sest seal on kõik kõigega seotud.

Aristarchus (c. 310 – c. 230 BC) proovis seda siiski, väites, et tegelikult tiirleb maakera ümber statsionaarse päikese. Ta uskus ka, et kinnistähed on teised päikesed, et universum on palju suurem kui arvati (ehkki kaasaegne seisukoht oli, et universumi mastaabis ei ole maakera suurem kui liivatera) ning, et maakera pöörleb ümber oma telje. Paraku ei suutnud Aristarchuse geotsentriiline teoria toetajaid leida, kuna see ei pidanud vastu vaatluslikule testile. Geotsentrilisest teoriast tuleneb nimelt loogilise paratamatusena, et tähtedel esineb maalt vaadates parallaks. See tähendab, et kui maakera koos astronoomiga teeb poolringi ümber päikese, siis kinnistähe näiv asukoht taevavölvil muutub, sest astronoom vaatleb teda teise nurga alt. Pange oma nimetissõrm näöst u 10 cm kaugusele, sulgege parem silm, seejärel avage see ning sulgege vasak silm ja te näete oma sõrme parallaksi selle näiva asukoha muutusena. Mõõtmised ei näidanud aga parallaksi olemasolu (sest maa trajektoori diameeter on palju lühem maa kaugusest tähtedest). Parallaksi suudeti esimest korda mõõta alles 1838, siis kui juba iga koolijüts uskus, et maakera tiirleb ümber päikese!

Ühte Aristotelese kosmoloogia olulist puudust nähti siiski kohe. Nimelt ei suuda Aristoteles seletada, miks osad planeedid teavavölvil vahest suunda muudavad ja

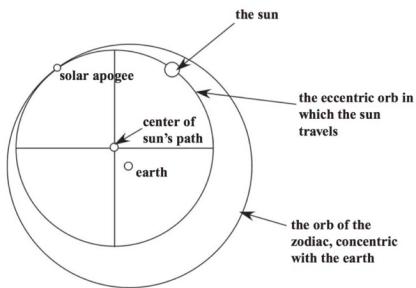


Figure 1.2: Ilma epitsükliteta ptolemailine mudel.

mõnda aega lausa vastupidises suunas liiguavad (retrogressioon). Kuna astronoomiat kasutasid põhiliselt astroloogid, siis pöörati planeetide liikumisele suurt tähelepanu. Lahenduseks ei olnud aga mitte suure teoria ümbertegemine või ümberlükkamine, vaid uue teaduse nõudmine, mis "päästaks fenomenid". Siin tuli appi Ptolemaios (c. AD 100 – c. 170), kes lõi matemaatilise mudeli, kus planeedid mitte lihtsalt ei liigu ringtrajektoori mõõda, vaid samal ajal teevad ka väiksemaid ringe ümber esimese suure ringjoone. Neid väiksemaid ringe kutsutakse epitsükliteks. See mudel suutis planeetide liikumist taevavõlvil piisavalt hästi ennustada, et astroloogide seltskond maha rahustada.

Ptolemaiosel ja tema järgijatel oli tegelikult mitu erinevat mudelit. Osad neist ei sisaldanud epitsükleid ja maakera ei asunud tema mudelites universumi keskel, vaid oli sellest punktist eemale nihutatud — nii et päike ei teinud ringe ümber maakera vaid ümber tühja punkti. Kuna leidus epitsüklitega mudel ja ilma epitsükliteta mudel, mis andsid identseid ennustusi, on selge, et Aristotelese teoria ja fenomenide päästmise mudelid on põhimõtteliselt erinevad asjad. Samal ajal, kui Aristoteles **seletas** maailma põhiolemust põhjuslike seoste jadana (mitte matemaatiliselt), **kirjeldas/ennustas** Ptolemaios sellesama maailma käitumist matemaatiliste (mitte põhjuslike) struktuuride abil.

Nii tekkis olukord, kus maailma mõistmiseks kasutati Aristotelese ühendteooriat, aga selle kirjeldamiseks ja tuleviku ennustamiseks hoopis ptolemailisi mudeliteid, mida keegi päriselt töeks ei pidanud ja mida hinnati selle järgi, kui hästi need "päästsid fenomene".

See toob meid Kopernikuse (1473 – 1543) juurde, kes teadusajaloolaste arvates vallandas 17. sajandi teadusliku revolutsiooni, avaldades raamatut, kus ta asetab päikese universumi keskele ja paneb maa selle ümber ringtrajektooril tiirlema. Kas Kopernikus tõrjus sellega kõrvale Aristotelese, Ptolemaiose või mõlemad? Tundub, et ta soovis kolmandat, suutis esimest ning tolleaegsete lugejate arvates üritas teha teist — ehk välja pakkuda alternatiivi ptolemailistele mudelitele, mis selleks ajaks olid muutunud väga keerukaks (aga ka samavõrra ennustustäpseks). Kuna Kopernikuse raamat läks trükki ajal, mil selle autor

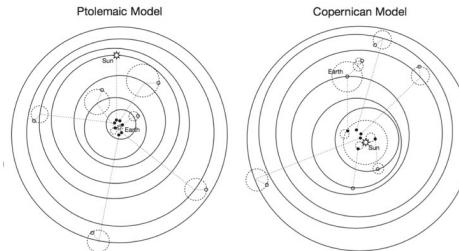


Figure 1.3: Ptolemaiose ja Kopernikuse mudelid on üllatavalt sarnased.

oli juba oma surivoodil, kirjutas sellele eessõna üks tema vaimulikust sõber, kes püüdis oodatavat kiriklikku pahameelt leevendada vihjates, et päikese keskele viimine on vaid mudeldamise trikk, millest ei tasu järeldada, et maakera ka tegelikult ümber päikese tiirleb (piibel räägib, kuidas jumal peatas taevavõlvil päikese, mitte maa). Ja kuna eessõna oli anonüümne, eeldasid lugejad, et selle kirjutas autor. Lisaks, kuigi Kopernikus tööstis päikese keskele, jäi ta planeetide ringikujuliste trajektooride juurde, mis tähendab, et selleks, et tema teooria fenomenide päästmisel hätta ei jäeks, oli ta sunnitud maad ja planeete mõõda epitsükleid ümber päikese liigutama. Kokkuvõttes oli Kopernikuse mudel umbes sama keeruline kui ptolemailikud mudelid ja selle abil tehtud ennustused planeetide liikumise kohta olid väiksema täpsusega. Seega, ennustava mudelina ei olnud sel suuri eeliseid.

Kopernikuse mudel suutis siiski ennustada mõningaid nähtusi (planeetide näiv heledus jõuab maksimumi nende lähimas asukohas maale), mida Ptolemaiose mudel ei ennustanud. See ei tähenda, et need fenomenid oleksid olnud vastuolus Ptolemaiose mudeliga. Lihtsalt, nende Ptolemaiose mudelisse sobitamiseks oli vaja osad mudeli parameetrid fikseerida nii-öelda suvalistele väärustustele. Seega Kopernikuse mudel töötas sellisel kujul, nagu see esitati, samas kui Ptolemaiose mudel vajas *post hoc* tuunimimst.

Kui vaadata Koperniku produkti teooriana, mitte mudelina, siis oli sellel küll selgeid eeliseid Aristotelese maailmateooria ees. Juba ammu oli nähtud komeete üle taevavõlv lendamas (mis Aristotelese järgi asusid kinnistähtede muutumatus sfääris), nagu ka supernova tekkimist ja kadu, ning enam ei olnud kaugel aeg, mil Galileo joonistas oma teleskoobist kraatreid kuu pinnal, näidates, et kuu ei saanud koosneda täiuslikust viiendast elemendist ja et sellel toimusid ilmselt sarnased füüsikalised protsessid kui maal. On usutav, et kui Kopernikus oleks jõudnud oma raamatule ise eessõna kirjutada, oleks tema teooria vastuvõtt olnud kiirem ja valulisem.

Chapter 2

Küsimused, mida statistika küsib

Statistika abil saab vastuseid järgmisi tele küsimustele:

- 1) kuidas näevad välja teie andmed ehk milline on just teie andmete jaotus, keskväärtus, varieeruvus ja koos-varieeruvus? Näiteks, mõõdetud pikkuste ja kaalude koos-varieeruvust saab mõõta korrelatsioonikordaja abil.
- 2) mida me peaksime teie andmete põhjal uskuma selle kohta, mis on päriselt? Näiteks, kui meie andmete põhjal arvutatud keskmise pikkus on 178 cm, siis kui palju on meil põhjust arvata, et tegelik populatsiooni keskmise pikkus > 185 cm?
- 3) mida ütleb statistilise mudeli struktuur teadusliku hüpoteesi kohta? Näiteks, kui meie poolt mõõdetud pikkuste ja kaalude koos-varieeruvust saab hästi kirjeldada kindlat tüüpi lineaarse regressioonimudeliga, siis on meil ehk tõendusmaterjali, et pikkus ja kaal on omavahel sellisel viisil seotud ja eelistatud peaks olema teaduslik teooria, mis just sellise seose tekkimisele bioloogilise mehanismi annab.
- 4) mida ennustab mudel tuleviku kohta? Näiteks, meie lineaarne pikkuse-kaalu mudel suudab ennustada tulevikus kogutavaid pikkuse andmeid. Aga kui hästi?

statistika peamine ülesanne on kvantifitseerida kõhedust, mida peaksime tundma vastates eeltoodud küsimustele.

Statistika ei vasta otse teaduslikele küsimustele ega küsimustele päris maailma kohta. Statistiklised vastused jäädvad alati kasutatud andmete ja mudelite piiridesse. Sellega seoses peaksime eelistama hästi kogutud rikkalikke andmeid ja paindlikke mudeliteid. Siis on lootust, et hüpe mudeli koefitsientidest päris maailma kirjeldamisse tuleb üle kitsama kuristiku. Bayesil on siin eelis, sest

osav statistik suudab koostöös teadlastega priori mudelisse piisavalt kasulikku infot koguda. Ühtlasi, mida paindlikum meetod, seda vähem automaatne on selle mõistlik kasutamine.

Jäta meelde

1. Statistika jagatakse kolme ossa: kirjeldav (summary), uuriv (exploratory) ja järel dav (inferential).
2. Kirjeldav statistika kirjeldab teie andmeid summaarsete statistikute abil.
3. Uuriv statistika püstitab valimi põhjal uusi teaduslikke hüpoteese, kasutades selleks põhiliselt graafilisi meetodeid.
4. Järel dav statistika kasutab formaalseid mudeliteid, et kontrollida uuriva statistika abil püsitatud hüpoteese. Järel dav statistika teeb valimi põhjal järel dusi statistilise populatsiooni kohta, milles see valim pärib.
5. Statistika põhjal tehtud järel dused on alati ebakindlad; ka siis kui need esitatakse punkthinnanguna parameetrväärtusele. Nii punkthinnangud kui intervall-hinnangud on lihtsustused: tegelik ebakindluse määär on n-dimensiooneline tõenäosuspilv, kus n on mudeli parameetrite arv.
6. Statistika põhiline ülesanne on kvantifitseerida ebakindlust, mis ümbritseb järel dava statistika abil saadud hinnanguid. Selle ebakindluse numbriline mõõt on tõenäosus, mis jäab 0 ja 1 vahel.
7. Tõenäosus omistab numbrilise väärtuse sellele, kui palju me usuksime hüpo teesi x kehtimisse, juhul kui me usuksime, et selle tõenäosuse arvutamiseks kasutatud statistilised mudelid vastavad tegelikkusele.
8. Ükski statistiline mudel ei vasta tegelikkusele ja mudelivaba statistikat ei ole olemas.

Chapter 3

Kuidas näevad välja teie andmed

Summaarsed statistikud

Summaarne statistik püüab iseloomustada teie valimit ühe numbri abil. Milliseid summaarseid statistikuid arvutada ja milliseid võltida, sõltub statistilisest mudelist, mis omakorda sõltub teie andmetest ja teie uskumustest andmeid genereeriva protsessi kohta.

Summaarse statistika abil iseloomustame

- tüüpilist valimi liiget (keskmise näitajad),
- muutuja sisest varieeruvust (standardhälve, mad jms),
- erinevate muutujate koos-varieeruvust (korrelatsioonikordaja)

Keskväärтused

Keskväärтust saab mõõta paaril tosinal erineval viisil, millest järgnevalt vaatleme kolme või nelja. Enne kui arvutama kukute, mõelge järele, miks te soovite keskväärтust teada. Kas teid huvitab valimi tüüpiline liige? Kuidas te sooviksite seda tüüpilisust defineerida? Kas valimi keskmise liikmena või valimi kõige arvukama liikmena? või veel kuidagi? See, millist keskväärтust kasutada, sõltub sageli andmejaotuse kujust. Sümmeetrilisi jaotusi on lihtsam iseloomustada ja mitmetipulised jaotused on selles osas kõige kehvemad.

Järgnevad nõuanded on rangelt soovituslikud:

density.default(x = andmed, adjust = 1)

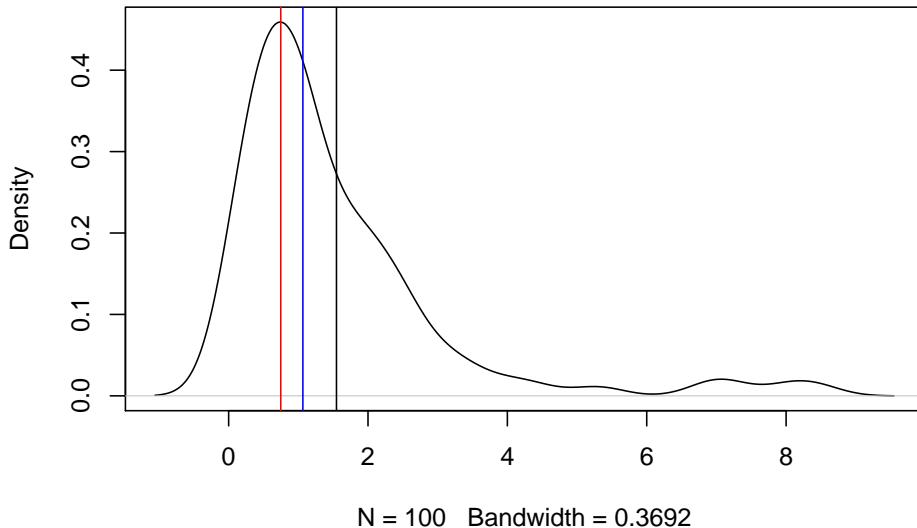


Figure 3.1: Simuleeritud lognormaaljaotusega andmed. Punane joon - mood; sinine joon - mediaan; must joon - aritmeetiline keskmise (mean). Milline neist vastab parimini teie intuitsiooniga nende andmete “keskväärtusest”? Miks?

- (1) Kui valim on normaaljaotusega (histogramm on sümmeetriseline), hinda tüüpilist liiget läbi aritmeetilise keskmise (mean).
- (2) Muidu kasuta mediaani (median). Kui valim on liiga väike, et jaotust hinnata (aga > 4), eelista mediaani. Mediaani saamiseks järjestatakse mõõdetud väärtsused suuruse järgi ja võetakse selle rea keskmise liige. Mediaan on vähem tundlik ekstreemsete väärtsuste (outlierite) suhtes kui mean.
- (3) Valimi kõige levinumat esindajat iseloomustab mood ehk jaotuse tipp. Seda on aga raskem täpselt määrata ja mitmetipulisel jaotusel on mitu moodi. Töötamisel posterioorsete jaotustega on mood sageli parim lahendus.

Muutuja sisene varieeruvus

Aritmeetilise keskmisega (*mean*) käib kokku standardhälve (SD). SD on sama ühikuga, mis andmed (ja andmete keskväärtus). Suhtelist standardhälvet kutsutakse variatsiooni koefitsiendiks (*Coefficient of Variation*) ja arvutatakse nii: $CV = \text{sd}(x)/\text{mean}(x)$. CV $\times 100\%$ annab varieeruvuse protsendina keskväärtusest.

Statistikute hulgas eelistatud andmete avaldamise formaat on mean (SD), mitte mean (+/- SD). 1 SD katab 68% normaaljaotusest, 2 SD – 96% ja 3 SD – 99%. Normaaljaotus langeb servades kiiresti, mis tähendab, et tal on peenikesed sabad ja näiteks 5 SD kaugusel keskmisest paikneb vaid üks punkt miljonist. Näiteks: inimeste IQ on normaaljaotusega, mean = 100, sd = 15. See tähendab, et kui sinu IQ = 115 (mis on enam-vähem ülikooli astujate keskmise IQ), siis on töenäosus, et juhuslikult kohatud inimene on sinust nutikam, 18% ((100% - 68%) / 2 = 18%).

Kui aga “tegelikul” andmejaotusel on “paks saba” (nagu eelmisel joonisel kujutatud andmetel) või esinevad outlierid, siis normaaljaotust eeldav mudel tagab ülehinnatud SD ja seega ülehinnatud varieeruvuse. Kui andmed saavad olla ainult positiivsed, siis $SD > \text{mean}/2$ viitab, et andmed ei sobi normaaljaotuse mudeliga (sest mudel ennustab negatiivsete andmete esinemist küllalt suure sagedusega).

Standardhälve on defineeritud ka mõnede teiste jaotuste jaoks peale normaaljaotuse (Poissoni jaotus, binoomjaotus). Funktsioon $sd()$ ja selle taga olev võrrand $sd = \sqrt{(\text{mean}(x) - x)^2/(n - 1)}$ on loodud normaaljaotuse tarbeks ja neid alternatiivseid standardhälbeid ei arvuta. Veelgi enam, igale jaotusele, mida me oskame integreerida, saab ka integraali abil õige katvusega standardhälbe arvutada. Seega tasub meeles pidada, et tavapärane viis standardhälbe arvutamiseks $sd()$ abil kehtib normaaljaotuse mudeli piirides ja ei kusagil mujal! Siiski, kui arvutada standardhälbe $sd()$ -ga, võib olla kindel, et jaotusest sõltumata hõlvavad 2 SD-d vähemalt 75% andmejaotusest. Kui andmed ei sobi normaaljaotusesse ja te ei ole rahul tulemusega, mille tõlgendus on niivärd ebakindel kui 75 protsentti kuni 96+ protsendi, võib pakkuda kahte alternatiivset lahendust:

Logaritmi andmed

Kui kõik andmeväärtused on positiivsed ja andmed on lognormaaljaotusega, siis logaritmimine muudab andmed normaalseks. Logaritmitud andmetest tuleks arvutada aritmeetiline keskmne ja SD ning seejärel mõlemad anti-logaritmida (näiteks, kui $\log_2(10) = 3.32$, siis antilogaritm sellest on $2^{3.32} = 10$). Sellisel juhul avaldatakse lõpuks geomeetriline keskmne ja multiplikatiivne SD algses lineaarses skaalas (multiplikatiivne SD = geom mean x SD; geom mean/SD). Geomeetriline keskmne on alati väiksem kui aritmeetiline keskmne. Lisaks on SD intervall nüüd asüümmeetriline ja SD on alati > 0 . See protseduur tagab, et 68% lognormaalsetest andmetest jäääb 1 SD vahemikku ning 96% andmetest jäääb 2 SD vahemikku.

Kui lognormaalsetele andmetele arvutada tavaline SD lineaarses skaalas kasutades $sd()$ funktsiooni, siis tuleb SD sageli palju laiem kui peaks ja hõlmab ka negatiivseid väärtsusi (pea meeles, et SD definitsiooni järgi jäääb 96% populatsioonist 2 SD vahemikku).

Sageli on aga negatiivsed muutuja vääritudused võimatuud (näiteks nädalas suitsetatud sigarettide arv).

Logaritmimise kaudu avaldatud multiplikatsiivse SD arvutamiseks kasutame enda kirjutatud funktsiooni multiplicative_sd(). Esiteks arvutame multiplikatiivse ja aditiivse sd lognormaalsetele andmetele, mida kujutasime eelmisel joonisel:

```

multiplicative_sd <- function(x) {
  x <- na.omit(x) #viskan välja NA-d (kui neid on)
  log_data <- log10(x) #logaritmin andmed
  log_mean <- mean(log_data) #keskmene logaritmitud andmetest
  log_sd <- sd(log_data) #SD logaritmitud andmetest
  geom_mean <- 10^log_mean #anti-logaritm annab geomeetrilise keskmise
  mult_sd <- 10^log_sd #anti-logaritm annab multiplikatiivse SD
  lower1 <- geom_mean/mult_sd #alumine SD piir
  upper1 <- geom_mean * mult_sd #ülemine SD piir
  lower2 <- geom_mean/(mult_sd^2) #alumine 2 SD piir
  upper2 <- geom_mean * (mult_sd^2) #ülemine 2 SD piir
  Mean <- mean(x) #aritmeetiline keskmene
  lower3 <- mean(x) - sd(x) #alumine additiivse SD piir
  upper3 <- mean(x) + sd(x) #ülemine additiivse SD piir
  lower4 <- mean(x) - sd(x)*2 #alumine additiivse 2 SD piir
  upper4 <- mean(x) + sd(x)*2 #ülemine additiivse 2 SD piir
  tibble(SD = c("multiplicative_SD",
               "multiplicative_2_SD",
               "additive_SD",
               "additive_2_SD"),
         MEAN = c(geom_mean,
                 geom_mean,
                 Mean,
                 Mean),
         lower = c(lower1,
                   lower2,
                   lower3,
                   lower4),
         upper = c(upper1,
                   upper2,
                   upper3,
                   upper4))
}

andmed <- rlnorm(100) #100 juhuslikku arvu lognormaaljaotusesest
multiplicative_sd(andmed) %>% knitr::kable()

```

SD	MEAN	lower	upper
multiplicative_SD	0.939	0.375	2.36
multiplicative_2_SD	0.939	0.149	5.91
additive_SD	1.448	-0.146	3.04
additive_2_SD	1.448	-1.740	4.64

Tavalise aritmeetilise keskmise asemel on meil nüüd geomeetriline keskmine. Võrdluseks on antud ka tavaline (aritmeetiline) keskmine ja (aditiivne) SD. Additiivne SD on selle jaotuse kirjeldamiseks selgelt ebaadekvaatne (vt jaotuse pilti ülalpool ja võrdle mulitplikatiivse SD-ga).

Kuidas aga töötab multiplikatiivne standardhälve normaaljaotusest pärit andmetega ($N=3$, $\text{mean}=100$, $\text{sd}=20$)? Kui multiplikatiivse sd rakendamine normaalsete andmete peal viiks katastroofini, siis poleks sel statistikul suurt kasutusruumi.

SD	MEAN	lower	upper
multiplicative_SD	108	92.8	126
multiplicative_2_SD	108	79.7	147
additive_SD	109	92.1	126
additive_2_SD	109	75.2	143

Nagu näha, on multiplikatiivse sd kasutamine normaalsete andmetega pigem ohutu (kui andmed on positiivsed). Arvestades, et additiivne SD on lognormaalsete andmete korral kõike muud kui ohutu ning et lognormaaljaotus on bioloogias üsna tavaline, eriti ensüümreaktsioonide ja kasvuprotsesside juures, on mõistlik alati kasutada multiplicative_sd() funktsiooni. Kui mõlema SD vääritud on sarnased, siis võib loota, et andmed on normaalsed ning saab refereede röömuks avaldada tavapärase additiivse SD.

kui $n < 10$, siis mõlemad SD-d alahindavad süsteematiselt tegelikku sd-d. Ettevaatust väikeste valimitega!

Vahest tekkib teil vajadus empiiriliselt määrrata, kas teie andmed on normaaljaotusega. Enne kui seda tegema asute, peaksite mõistma, et see, et teie valim ei ole normaalne, ei tähenda automaatselt, et populatsioon, millest see valim tõmmati, ei oleks normaaljaotusega. Igal juhul, valimiandmete normaalsuse määramiseks on kõige mõistlikum kasutada qq-plotti. QQ-plot (kvantiil-kvantiil plot) võrdleb andmete jaotust ideaalse normaaljaotusega andmepunkti haaval. Kui empiiriline jaotus kattub referentsjaotusega, siis on tulemuseks sirgel paiknevad punktid. Järgneval qq plotti on näha, mis juhtub, kui plottida lognormaalseid andmeid normaaljaotuse vastu:

```
qqPlot(andmed)
```

```
#> [1] 66 65
```

Nüüd joonistame qq-ploti logaritmitud andmetele.

```
qqPlot(log(andmed))
```

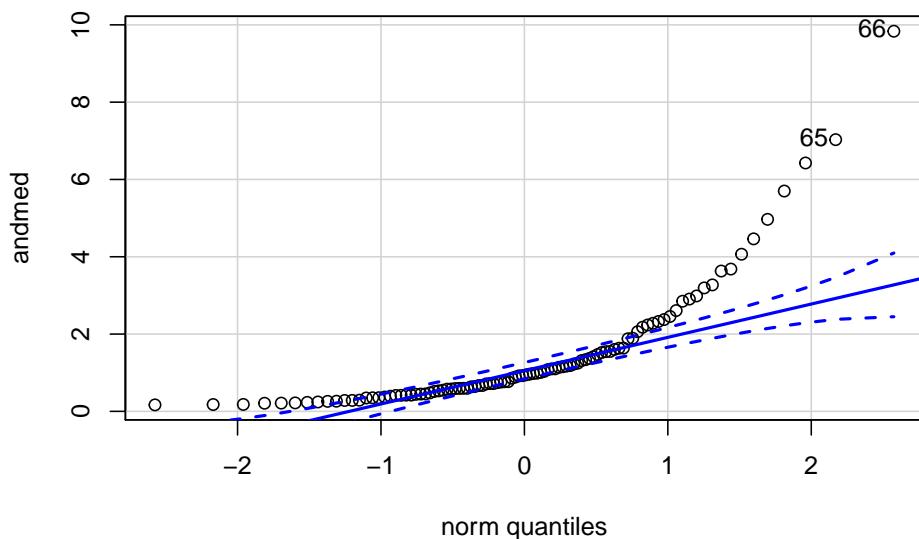


Figure 3.2: QQ-plot lognormaalsetele andmetele. Plotil võrreldakse lognormaalsete andmete jaotust referentsjaotusega, milleks on antud juhul normaaljaotus. Punased katkendjooned annavad standardveapõhise usaldusvahemiku (arvutatud simuleeritud juhuvalimist normaaljaotusega referentspopulatsioonist), millesse peaks jääma enamus andmepunkte juhul kui andmepunktid pärineksid normaaljaotusest.

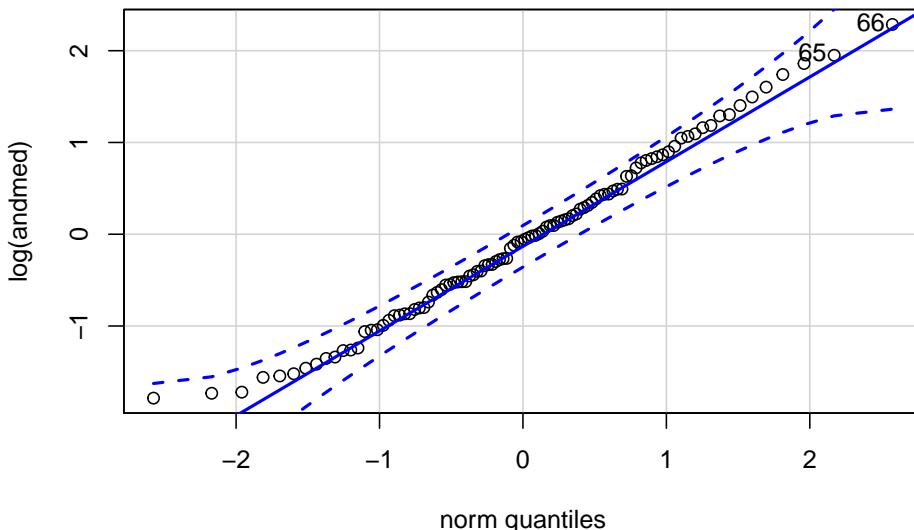


Figure 3.3: QQ-plot normaalsetele andmetele (logaritmitud lognormaalsed andmed).

```
#> [1] 66 65
```

Pole kahtlust, andmed on logaritmitud kujul normaaljaotusega.

`qqPlot()` võimaldab võrrelda teie andmeid ükskõik millise R-is definieritud jaotusega (`?car::qqPlot`).

Normaaljaotuse kindlakstegemiseks on loodud ka peotäis sageduslikke teste, mis annavad väljundina p väärtsuse. Nende kasutamisest soovitame siiski hoiduda, sest tulemused on sageli ebakindlad, eriti väikestel ja suurtel valimitel. Mõistlikum on vaadata kõikide andmepunktide plotti normaaljaotuse vastu, kui jällitada ühte numbrit (p), mille väärthus, muuseas, monotooniliselt langeb koos valimi suuruse kasvuga.

Iseloomusta andmeid algses skaalas: mediaan (MAD)

MAD — median absolute deviation — on vähem tundlik outlierite suhtes ja ei eelda normaaljaotust. Puuduseks on, et MAD ei oma tõlgendust, mille kohaselt ta hõlmaks kindlat protsendi populatsiooni või valimi andmejaotusest. Seevastu sed puhul võime olla kindlad, et isegi kõige hullemaga jaotuse korral jäavat vähemalt 75% andmetest 2 SD piiridesse.

Lognormaalsete andmetega:

```
mad(andmed, constant = 1); sd(andmed); mad(andmed)
#> [1] 0.522
#> [1] 1.59
#> [1] 0.774

mad(log10(andmed), constant = 1); sd(log10(andmed)); mad(log10(andmed))
#> [1] 0.275
#> [1] 0.399
#> [1] 0.408
```

`mad` = $\text{median}(\text{abs}(\text{median}(x) - x))$, mida on väga lihtne mõista. Samas R-i funktsioon `mad()` korruatab default-ina `mad`-i läbi konstandiga 1.4826, mis muudab `mad()`-i tulemuse võrreldavaks `sd`-ga, tehes sellest `sd` robustse analoogi. Robustse sellepärast, et `mad`-i arvutuskäik, mis sõltub mediaanist, mitte aritmeetilisest keskmisest, ei ole tundlik outlierite suhtes. Seega, kui tahate arvutada `mad`-i, siis fikseerige `mad()` funktsionis argument *constant* ühele.

Ära kunagi avalda andmeid vormis: mean (MAD) või median (SD).
Korrektne vorm on mean (SD) või median (MAD).

Veel üks viis andmejaotuse summeerimiseks on kasutada kvantiile. Siin saame me tüüpiliselt rohkem kui ühe numbri, aga sageli on selline viis informatiivsem, kui ühenumbrilised summaarsed statistikud. Funktsioon `quantile` võimaldab valida, millisid kvantiile soovite näha. Järgnevas koodist saame teada, millisest vektori "andmed" väärustest allapoole jääb 2.5%, 25%, 50%, 75% ja 95% väärusti.

```
quantile(andmed, c(0.025, 0.25, 0.5, 0.75, 0.95))
#> 2.5% 25% 50% 75% 95%
#> 0.194 0.471 0.935 1.634 4.488
```

Muutujate koosvarieeruvus

Andmete koos-varieeruvust mõõdetakse korrelatsiooni abil. Tulemuseks on üks number - korrelatsionikordaja r , mis varieerub -1 ja 1 vahel.

- $r = 0$ – kahte tüüpi mõõtmised (x =pikkus, y =kaal) samadest mõõteobjektidest varieeruvad üksteisest sõltumatult.
- $r = 1$: kui ühe muutuja väärustus kasvab, kasvab ka teise muutuja väärustus alati täpselt samas propportsioonis.
- $r = -1$: kui ühe muutuja väärustus kasvab, kahaneb teise muutuja väärustus alati täpselt samas propportsioonis.

Kui r on -1 või 1, saame me x väärust teades täpselt ennustada y väärustuse (ja vastupidi, teades y väärust saame täpselt ennustada x väärustuse).

Kuidas tõlgendame aga tulemust $r = 0.9$? Mitte kuidagi. Selle asemel tõlgendame

$r^2 = 0.9^2 = 0.81$ – mis tähendab, et x-i varieeruvus suudab seletada mitte rohkem kui 81% y varieeruvusest ja vastupidi, et Y-i varieeruvus suudab seletada 81% X-i varieeruvusest.

Korrelatsiooni saab mõõta mitmel viisil (`?cor.test, method=`). Kõige levinum on Pearsoni korrelatsioonikoefficients, mis eeldab, (i) et me mõõdame pidevaid muutujaid, (ii) juhuvalimit, (iii) et populatsiooniandmed on normaaljaotusega ja (iv) et igal mõõteobjektil on mõõdetud 2 omadust (pikkus ja kaal, näiteks). Tuntuim alternatiiv on mitteparametriseline Spearmani korrelatsioon, mis ei eelda andmete normaaljaotust ega seda, et mõõdetakse pidevaid suurusi. Kui Pearsoni korrelatsiooni eeldused on täidetud ja te kasutate siiski Spearmani korrelatsiooni, siis langeb teie arvutuse efektiivsus ca. 10% võrra.

```
cor(iris$Sepal.Length, iris$Sepal.Width, use = "complete.obs")
#> [1] -0.118
```

Korrelatsioonikordaja väärthus sõltub mitte ainult andmete koosvarieeruvusest vaid ka andmete ulatusest. Suurema ulatusega andmed X ja/või Y teljel annavad keskelt läbi 0-st kaugemal oleva korrelatsioonikordaja. Selle pärast sobib korrelatsioon halvasti näiteks korduskatsete kooskõla mõõtmiseks.

Lisaks, korrelatsioonikordaja mõõdab vaid andmete lineaarsel koos-varieeruvust: kui andmed koos-varieeruvad mitte-lineaarselt, siis võivad ka väga tugevad koos-varieeruvused jäada märkamatuks.

Moraal seisneb selles, et enne korrelatsioonikordaja arvutamist tasub alati plotida andmed, et veenduda võimaliku seose lineaarsuses. Lineaarsuse piudumine andmete koosvarieeruvuse mustris tähendab, et korrelatsioonikordaja tuleb eksitav.

Korrelatsioonikordaja mõõdab pelgalt määra, mil üks muutuja muutub siis, kui teine muutuja muutub. Seega ei ole suurt mõtet arvutada korrelatsioonikordajat juhul kui me teame ette seose olemasolust kahe muutuja vahel. Näiteks, kui sama entiteeti mõõdetakse kahel erineval viisil, või kahes korduses, või kui esimene muutuja arvutatakse teise muutuja kaudu.

Kõik summaarsed statistikud kaotavad suure osa teie andmetes leiduvast infost – see kaotus on õigustatud ainult siis, kui teie poolt valitud statistik iseloomustab hästi andmete sügavamat olemust (näiteks tüüpilist mõõtmistulemust või andmete varieeruvust).

Korrelatsioonimaatriksi saab niimoodi:

```
# numeric columns only!
# the following gives cor matrix with
# frequentist correction for multiple testing:
# print(psych::corr.test(iris[-5]))
# only numeric cols allowed! Hence -Species
knitr::kable(cor(iris[,-5]))
```

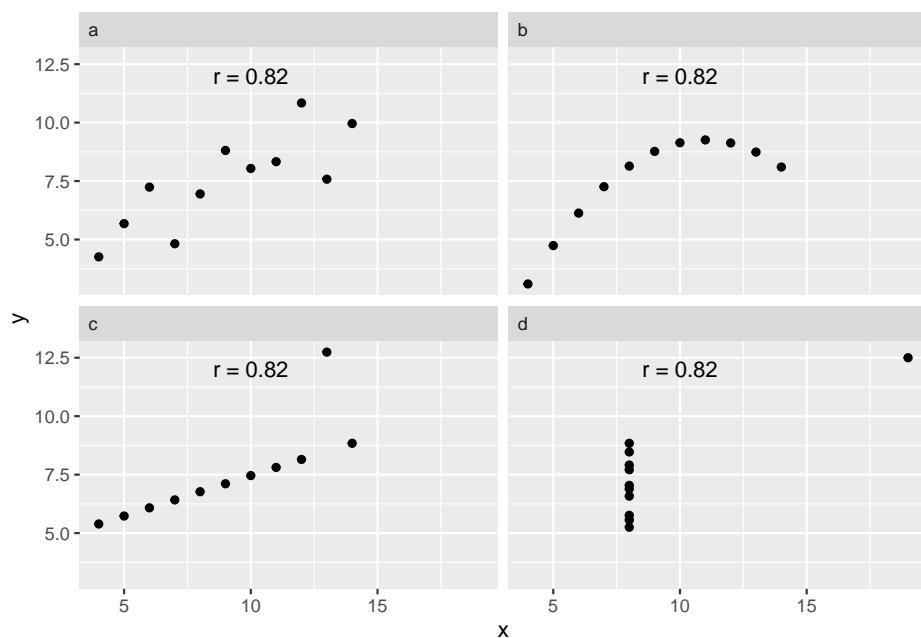


Figure 3.4: Anscombe'i kvartett illustreerib korrelatsioonikordaja lineaarset olemust: neli andmestikku annavad identse korrelatsioonikordaja (Pearsons' r), ehkki tegelikud seosed andmete vahel on täiesti erinevad.

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Sepal.Length	1.000	-0.118	0.872	0.818
Sepal.Width	-0.118	1.000	-0.428	-0.366
Petal.Length	0.872	-0.428	1.000	0.963
Petal.Width	0.818	-0.366	0.963	1.000

Chapter 4

Lineaarsed mudelid

```
library(tidyverse)
library(ggthemes)
library(broom)
library(modelr)
library(viridis)
```

4.1 Sirge võrrand

Oletame, et me mõõtsime N inimese pikkuse cm-s ja kaalu kg-s ning meid huvitab, kuidas inimeste pikkus sõltub nende kaalust. Lihtsaim mudel pikkuse sõltuvusest kaalust on pikkus = kaal (formaliseeritult: $y = x$) ja see mudel ennustab, et kui Juhani kaal = 80 kg, siis Juhan on 80 cm pikkune. Siin on pikkus muutuja, mille väärust ennustatakse (y) ja kaal muutuja x , mille väärustete põhjal ennustatakse pikkuusi. Muidugi, sama hästi võiksime ennustada kaalu pikkuste põhjal, ja kumma ennustuse valime sõltub siinkohal eeskätt meie teaduslikest huvidest.

Veelgi enam, sama mudel ennustab, et kui Juhani kaal = 80 tonni, siis on Juhan samuti 80 cm pikkune (eeldusel, et me avaldame x muutuja tonnides ja y muutuja sentimeetrites). Seega, mudeli ennustuse täpsus sõltub ühikutest, milles me andmed mudelisse sisse anname.

```
#Genereerime andmed: x = pikkus ja y = kaal:
x <- 0:100
y <- x
```

Selle mudeli saame graafiliselt kujutada nii:

```
plot(y ~ x,
      type = "l",
```

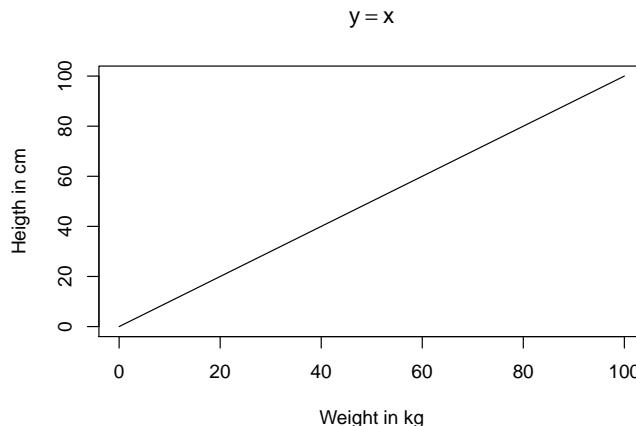


Figure 4.1: Lihtne mudel $y \sim x$, mille lõikepunkt = 0 ja tõus = 1.

```
xlab = "Weight in kg",
ylab = "Height in cm",
main = bquote(y == x))
```

Üldistatult, mudeli keeles tähistame me seda muutujat, mille väärtsust me ennustame, Y-ga ja seda muutujat, mille väärtsuse põhjal me ennustame, X-ga.

Sirge mudeli lihtsaim matemaatiline formalism on $Y = X$. See on äärmiselt jäik mudel: sirge, mille asukoht on rangelt fikseeritud. Sirge lõikab y telje alati 0-s (mudeli keeles: sirge *intercept* ehk lõikepunkt Y teljel = 0) ja tema tõusunurk saab olla ainult 45 kraadi (mudeli keeles *slope* ehk tõus = 1). Selle mudeli jäikus tuleneb sellest, et temas ei ole parameetreid, mille väärtsusi me saaksime vabalt muuta ehk tuunida.

Mis juhtub, kui me lisame mudelisse konstanti, mille liidame x-i väärustustele?

$$y = a + x$$

See konstant on mudeli parameeter, mille väärtsuse võime vabalt valida. Järgnevalt anname talle väärtsuse 30 (ilma konkreetse põhjuseta).

```
x <- 0:100
a <- 30
y <- a + x

plot(y ~ x, xlim = c(0, 100), ylim = c(0, 150), type = "l",
     main = bquote(y == a + x))
abline(c(0, 1), lty = 2)
```

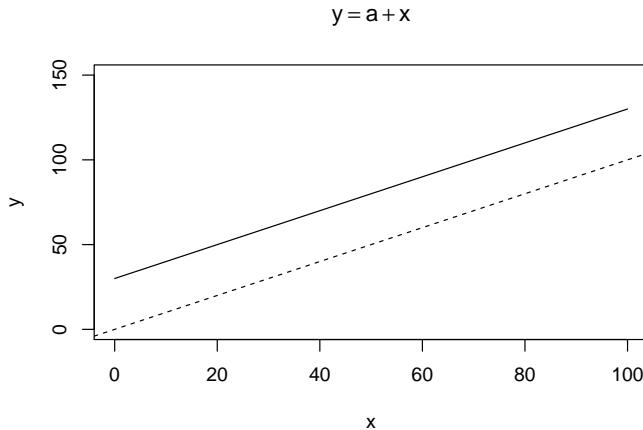


Figure 4.2: Lineaarne mudel, mille lõikepunkt = 30 ja tõus = 1. Katkendjoon, lõikepunkt = 0. Pidevjoon, lõikepunkt = 30

Meie konstant a määrab y väärtsuse, kui $x = 0$, ehk sirge lõikepunkt y teljel. Teisisõnu, a = mudeli lõikepunkt (*intercept*).

Mis juhtub, kui me mitte ei liida, vaid korrutame x -i konstandiga?

$$y = b \times x$$

Jällegi, me anname mudeli parameetrile b suvalise väärtsuse, 3.

```
x <- 0:200
b <- 3
y <- b * x

plot(y ~ x, xlim = c(0, 100), ylim = c(0, 100), type = "l", main = bquote(y == b %% x))
abline(c(0, 1), lty = 2)
```

Nüüd muutub sirge tõusunurk, ehk kui palju me ootame y -t muutumas, kui x muutub näiteks ühe ühiku võrra. Kui $b = 3$, siis x -i tõustes ühe ühiku võrra suureneb y kolme ühiku võrra. Proovi järgi, mis juhtub, kui $b = -3$.

Selleks, et sirget kahes dimensioonis vabalt liigutada, piisab kui me kombineerime eelnevad näited ühte:

$$y = a + b \times x$$

Selleks lisame mudelisse kaks parameetrit, lõikepunkt (a) ja tõus (b). Kui $a = 0$ ja $b = 1$, saame me eelpool kirjeldatud mudeli $y = x$. Kui $a = 102$, siis sirge lõikab y -telge väärtsel 102. Kui $b = 0.8$, siis x -i tõustes 1 ühiku võrra tõuseb y -i väärthus 0.8 ühiku võrra. Kui $a = 100$ ja $b = 0$, siis saame sirge, mis

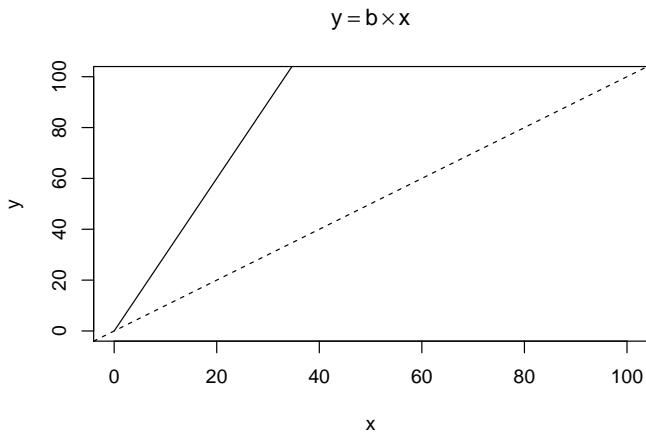


Figure 4.3: Lineaarne mudel, mille lõikepunkt = 0 ja tõus = 3. Katkendjoon, tõus = 1. Pidevjoon, tõus = 3.

on paraleeline x-teljega ja lõikab y-telge väärtsusele 100. Seega, teades a ja b väärtsusi ning omistades x-le suvalise meid huvitava väärtsuse, saab ennustada y-i keskmist väärust sellel x-i väärtsusel. Näiteks, olgu andmete vastu fititud mudel pikkus(cm) = 102 + 0.8 * kaal(kg) ehk

$$y = 102 + 0.8 \times x$$

Omistades nüüd kaalule väärtsuse 80 kg, tuleb mudeli poolt ennustatud keskmise pikkus $102 + 0.8 * 80 = 166$ cm. Iga kg lisakaalu ennustab mudeli kohaselt 0.8 cm võrra suuremat pikkust.

```
a <- 102
b <- 0.8
x <- 0:100
y <- a + b * x

plot(y ~ x, xlab = "Weight in kg", ylab = "Height in cm", ylim = c(50, 200), type = "l")
```

See mudel ennustab, et 0 kaalu juures on pikku 102 cm, mis on rumal, aga mudelite puhul tavalline olukord. Sellel olukorral on mitmeid põhjusi:

- Me tuunime mudelit andmete peal, mis ei sisalda 0-kaalu.
- Meie valimiandmed ei peegelda täpselt inimpopulatsiooni.
- Sirge mudel ei peegelda täpselt pikkuse-kaalu suhteid vahemikus, kus meil on reaalseid kaaluandmeid; ja ta teeb seda veelgi vähem seal, kus meil mõõdetud kaalusid ei ole.

Seega pole mõtet imestada, miks mudeli intercept meie üle irvitab.

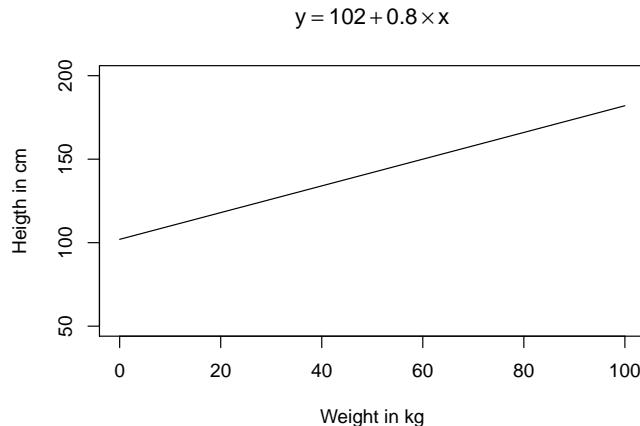


Figure 4.4: Lineaarne mudel, millel on tuunitud nii löikepunkt kui tōus.

Kahe parameetriga sirge mudel ongi see, mida me fitime kahedimensiooniliste andmetega.

Näiteks nii, kasutame R-i "iris" andmesetti:

```
# Fit a linear model and name the model object as m
m <- lm(Sepal.Length ~ Petal.Length, data = iris)

# Make a scatter plot, colored by the var called "Species"
# Draw the fitted regression line from m
augment(m, iris) %>%
  ggplot(aes(Petal.Length, Sepal.Length, color = Species)) +
  geom_point() +
  geom_line(aes(y = .fitted), color = 1) +
  labs(title = "Sepal.Length ~ Petal.Length") +
  scale_color_viridis(discrete = TRUE)
```

Mudeli fittimine tähdab siin lihtsalt, et sirge on 2D ruumi asetatud nii, et see oleks võimalikult lähedal kõikidele punktidele.

Oletame, et meil on n andmepunkti ja et me fitime neile sirge. Nüüd plotime fititud sirge koos punktidega ja tömbame igast punktist mudelsirgeni joone, mis on paraleelne y-teljega. Seejärel mõõdame nende n joone pikkused. Olgu need pikkused a, b, \dots i. $\text{lm}()$ funktsioon fitib sirge niimoodi, et summa $a^2 + b^2 + \dots + i^2$ oleks minimaalne. Seda kutsutakse vähimruutude meetodiks.

Mudeli koefitsientide väärtsused saame kasutades funktsiooni `coef()`:

```
coef(m)
#> (Intercept) Petal.Length
#>        4.307      0.409
```

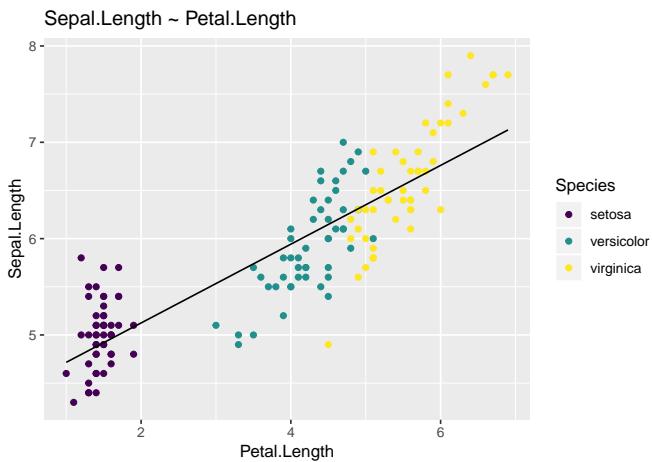


Figure 4.5: Fititud mudel, kus muutuja Petal.Length järgi ennustatakse muutuja Sepal.Length väärtsusi.

Siin a = (Intercept) ja b = Petal.Length ehk 0.41.

4.2 Ennustus lineaarsest mudelist

Anname x -le rea väärtsusi, et ennustada y keskmisi väärtsusi nendel x -i väärustel. Siin me ennustame y (Sepal_length) keskväärtusi erinevatel x -i (Petal_length) väärustel, mitte individuaalseid Sepal_length väärtsusi. Me kasutame selleks deterministlikku mudelit kujul $\text{Sepal_length} = a + b * \text{Petal_length}$. Hiljem õpime ka bayesiaanlike meetoditega individuaalseid Sepal_length-e ennustama.

Järgnev kood on sisuliselt sama, millega me üle-eelmisel plotil joonistasime mudeli $y = a + bx$. Me fikseerime mudeli koefitsiendid fititud irise mudeli omadega ja anname Petal_length muutujale 10 erinevat väärust originaalse muutuja mõõtmisvahemikus. Aga sama hästi võiksime ekstrapoleerida ja küsida, mis on oodatav Sepal_length, kui Petal_length on 100 cm? Sellele küsimusele on ebareaalne vastus, aga mudel ei tea seda. Proovi, mis vastus tuleb.

```
## Genereerime uued andmed Petal.Length vahemikus
Petal_length <- seq(min(iris$Petal.Length),
                      max(iris$Petal.Length),
                      length.out = 10)
## Võtame mudeli koefitsendid
a <- coef(m)[1]
b <- coef(m)[2]
## Kasutades mudeli koefitsente genereerime Sepal_length väärused
Sepal_length <- a + b * Petal_length
```

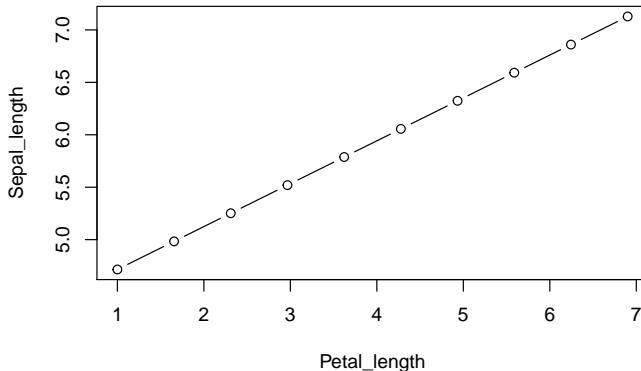


Figure 4.6: Siin ennustasime kümme y väärust x väärustuse põhjal..

```
plot(Sepal_length ~ Petal_length, type = "b")
```

Mudelist saab kahte tüüpi ennustusi: interpolatsioone (ennustus on samas skaalas, mis andmed, mille peal mudel fititi) ja ekstrapoleerimine (ennustus jäab väljaspoole andmeid). Sõltuvalt erialast me kas vaatame ekstrapoleerimisele viltu või leiame, et selline tegevus on otsesõnu keelatud.

Mudelist saab kahte tüüpi ennustusi: (1) saame ennustada Y keskmist väärustust X-i konkreetsel vääratusel ja (2) saame ennustada individuaalseid Y väärustusi X-i konkreetsel vääratusel. Viimase kohta vt ptk...

4.3 Neli möistet

Mudelis $y = a + bx$ on x ja y muutujad, ning a ja b on parameetrid. Muutujate väärused fikseeritakse andmetega, parameetrid fititakse andmete põhjal. Fititud mudel valib kõikide võimalike seda tüüpi mudelite hulgast välja täpselt ühe unikaalse mudeli ja ennustab igale x -i väärusele vastava kõige tõenäolisema y vääruse (y keskväärtuse sellel x -i vääratusel).

- Y — mida me ennustame (*dependent variable, predicted variable*).
- X — mille põhjal me ennustame (*independent variable, predictor*).
- Muutuja (variable) — iga asi, mida me valmis mõõdame (X ja Y on kaks muutujat). Muutujal on sama palju fikseeritud väärusti kui meil on selle muutuja kohta mõõtmisandmeid.
- Parameeter (parameter) — mudeli koefitsient, millele võib omistada suvalisi väärustusi. Parameetreid tuunides fitime mudeli võimalikult hästi sobituma

andmetega.

Mudel on matemaatilise formalism, mis püüab kirjeldada füüsikalist protsessi. Statistilise mudeli struktuuris on komponent, mis kirjeldab ideaalseid ennustusi (nn protsessi mudel) ja eraldi veakomponent (ehk veamudel), mis kirjeldab looduse varieeruvust nende ideaalsete ennustuste ümber. Mudeli koostisosad on (i) muutuja, mille väärtsus ennustatakse, (ii), muutuja(d), mille väärtsuste põhjal ennustatakse, (iii) parameetrid, mille väärtsused fititakse ii põhjal ja (iv) konstandid.

4.4 Mudeli fittimine

Mudelid sisaldavad nii (1) matemaatilisi struktuure, mis määradavad mudeli tüübi, kui (2) parameetrid, mida saab andmete põhjal tuunida, niiviisi täpsustades mudeli kuju ehk paiknemist matemaatlises ruumis. Näiteks võrrand $y = a + bx$ määrab mudeli, kus $y = x$ on see struktuur, mis tagab, et mudeli tüüp on sirge, ning a ja b on parameetrid, mis määradavad sirge asendi. Seevastu struktuur $y = x+x^2$ tagab, et mudeli $y = a+b_1x+b_2x^2$ tüüp on parabool, ning parameetrite a , b_1 ja b_2 väärtsused määradavad selle parabooli täpse kuju. Ja nii edasi.

Mudeli parameetrite tuunimist nimetatakse mudeli fittimiseks. Mudelit fittides on eesmärk saavutada antud tüüpi mudeli maksimaalne sobivus andmetega (kus “andmed” hõlmavad nii valimiandmeid kui taustateadmisi). Sellele tegevusele annab mõtte meie lootus, et mudeli tüüp kajastab mingit looduslike toimuvat protsessi, mis meile teaduslikku huvi pakub. Ning, kuigi mudeli fit maksimeeritakse mudeli tüübi kohta, püüab see andmete vaatenurgast vaadatuna olla optimaalne, mitte maksimaalne (vt järgmine peatükk mudeli üle- ja alafittimisest). Kahjuks ei ole selline optimaalsus kuigi hästi matemaatilisse vormi valata, ega ka mingi (pool)automaatse meetodiga empiiriliselt kontrollitav. Siin on tegu pigem teadlase sooviga, mille filosoofiline eeldus on, et meie andmetes on peidus nii andmeid genereeriva loodusliku protsessi üldine olemus (essents), kui juhuslik müra ehk valimiviga, ning et mudeli üldine kuju (sirge, parabool, jms) on juhtumisi sobiv just selleks, et neid kahte omavahel lahku ajada.

Lineraarse mudeli parima sobivuse andmetega saab tagada kahel erineval viisil: (i) vähimruutude meetod (Legendre, 1805; Gauss, 1809) mõõdab y telje suunaliselt iga andmepunkti kauguse mudeli ennustusest, võtab selle kauguse riutu, summeerib kauguste riitudud ning leib sirge asendi, mille korral see summa on minimaalne; (ii) Bayesi teoreem (Laplace, 1774) annab väheinformatiivse priori korral praktiliselt sama fiti. Olulise erinevusena võtab vähimruutude meetod arvesse ainult valimiandmed, samas kui Bayesi teoreemi kasutades fitime mudeli koefitsiente nii valimiandmete kui taustateadmiste peal (vt 8. ptk).

Hea mudel on

1. Võimalikult lihtsa struktuuriga, mille põhjal on veel võimalik teha järeldusi protsessi kohta, mis genereeris mudeli fittimiseks kasutatud andmeid;
2. Sobitub piisavalt hästi andmetega (eriti uute andmetega, mida ei kasutatud selle mudeli fittimiseks), et olla relevantne andmeid genereeriva protsessi kirjeldus;
3. Genereerib usutavaid simuleeritud andmeid.

Sageli fititkse samade andmetega mitu erinevat tüüpi mudelit ja püütakse otustada, milline neist vastab kõige paremini eeltoodud tingimustele. Näiteks, kui sirge suudab kaalu järgi pikkust ennustada paremini kui parabool, siis on sirge mudel paremas kooskõlas teadusliku hüpoteesiga, mis annaks mehhanismi protsessile, mille käigus kilode lisandumine viiks laias kaaluvahemikus inimeste pikkuse kasvule ilma, et pikkuse kasvu tempo kaalu tõustes langeks. Samas, see et me oleme oma andmeid fittinud n mudeliga ja otsustanud, et mõned neist on paremad kui teised, ei tähenda, et mõni meie mudelitest oleks hea ka võrdluses tegeliku looduslike valitseva olukorraga. Mudelid on pelgalt matematilised formalismid, mis võivad, aga kindlasti ei pea, kajastama füüsikalist maailma, ja meie mudelitevalik sõltub meile jõukohasest matemaatikast. Siinkohal ei tasu unustada, et matemaatika kirjeldab eelkõige abstraktseid mustreid, mitte otse füüsikalist maailma.

See, et teie andmed sobivad hästi mingi mudeliga, ei tähenda automaatselt, et see fakt oleks teaduslikult huvitav. Mudeli parameetrid on mõtekad mudeli matemaatilise kirjelduse kontekstis, aga mitte tingimata suure maailma põhjusliku seletamise kontekstis. Siiski, kui mudeli matemaatiline struktuur loodi andmeid genereeriva loodusliku protsessi olemust silmas pidades, võib mudeli koefitsientide uurimisest selguda olulisi tõsiasju suure maailma kohta.

Mudeli fittimine: X ja Y saavad oma väärtsused otse andmetest; parameetrid võivad omandada ükskõik millise väärtsuse.

Fititud mudelist ennustamine: X-le saab omistada ükskõik millise väärtsuse; parameetrite väärtsused on fikseeritud; Y väärthus arvatakse mudelist.

4.4.1 Üle- ja alafittimine

Osad mudelite tüübide on vähem paindlikud kui teised (parameetreid tuunides on neil vähem liikumisruumi). Kuigi sellised mudelid sobituvad halvemini andmetega, võivad need ikkagi paremini kui mõni paindlikum mudel välja tuua andmete peidetud olemuse. Statistiline mudeldamine eeldab, et me usume, et meie andmetes leidub nii müra (mida mudel võiks ignoreerida), kui signaal (mida mudel püüab tabada). Ilma signaalita süsteemi poleks arusaadavatel põhjustel mõtekas mudeldada ja ilma mürata süsteemi mudel tuleks ilma varieeruvuse (vea) komponendita, ehk deterministlik. Kuna mudeli jaoks näeb müra samamoodi välja kui signaal, on iga mudel kompromiss üle- ja alafittimise vahel. Me lihtsalt

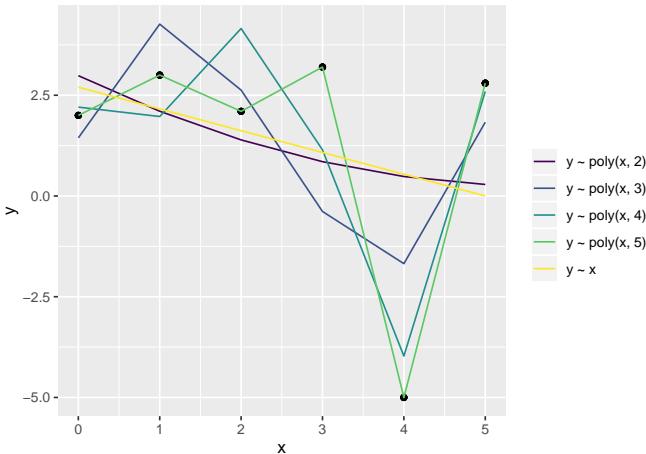


Figure 4.7: Kasvava paindlikusega polünoomsed mudelid.

loodame, et meie mudel on piisavalt jäik, et mitte liiga palju müra modelleerida ja samas piisavalt paindlik, et piisaval määral signaali tabada.

Üks kõige jäigemaid muudeleid on sirge, mis tähendab, et sirge mudel on suure töönäosusega alafittitud. Keera sirget kuipalju tahad, ikka ei sobitu ta enamiku andmekogudega. Ja need vähesed andmekogud, mis sirge mudeliga sobivad, on genereeritud teatud tüüpiliste lineaarsete protsesside poolelt. Sirge on seega üks kõige paremini tölgendatavaid muudeleid. Teises äärmuses on polünoomsed mudelid, mis on väga paindlikud, mida on väga raske tölgendada ja mille puhul esineb suur mudeli ülefittimise oht. Ülefittitud mudel järgib nii täpselt valimiandmeid, et sobitub hästi valimis leiduva juhusliku müraga ning seetõttu sobitub halvasti järgmiste valimiga samast populatsioonist (igal valimil on oma juhuslik müra). Üldiselt, mida rohkem on mudelis tuunitavaid parameetreid, seda paindlikum on mudel, seda kergem on seda valimiandmetega sobitada ja seda raskem on seda tölgendada. Veelgi enam, alati on võimalik konstrueerida mudel, mis sobitub täiuslikult kõikide andmepunktidega (selle mudeli parameetrite arv = N). Selline mudel on täpselt sama informatiivne kui andmed, mille põhjal see fititi — ja täiesti kasutu.

```
#> Warning: `cols` is now required.
#> Please use `cols = c(preds)`
```

Vähimruutude meetodil fititud muudeleid saame võrrelda AIC-i näitaja järgi. AIC - Akaike Informatsiooni Kriteerium - vaatab mudeli sobivust andmetega ja mudeli parameetrite arvu. Väikseim AIC tähitab parimat fitti väikseima parameetrite arvu juures (kompromissi) ja väikseima AIC-ga mudel on eelistatuim mudel. Aga seda ainult võrreldud mudelite hulgas. AIC-i absoluutväärus ei loe - see on suhteline näitaja.

model_formula	aic
y ~ x	35.0
y ~ poly(x, 2)	37.0
y ~ poly(x, 3)	36.1
y ~ poly(x, 4)	32.5
y ~ poly(x, 5)	-Inf

AIC näitab, et parim mudel on mod_e4. Aga kas see on ka kõige kasulikum mudel? Mis siis, kui 3-s andmepunkt on andmesisestaja näpuviga?

Ülefittimise vältimiseks kasutavad Bayesi mudelid informatiivseid prioreid, mis välistavad ekstreemsed parameetrväärtused. Vt <http://elevanth.org/blog/2017/08/22/there-is-always-prior-information/>

4.5 Lineaarse regressiooni eeldused

Matemaatilised eeldused:

1. Lineaarsus: ennustus Y-muutujale on lineaarne funktsioon prediktoritest $Y = \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$, ehk ekvivalentelt: kõikidel X-i väärustel keksmine residuaal = 0.
2. homoskedastilisus ehk konstantne Y-muutuja suunaline varieeruvus kõigil X-i väärustel. See tähendab ühtlasi residuaalide konstantset varieeruvust.
3. Normaalsus - residuaalid on normaaljaotusega, ehk Y-muutuja on normaaljaotusega kõgil X-i väärustel.
4. Sõltumatus - residuaalide väärused ei ole omavahel korreleeritud.
5. X-muutuja suunal puudub mõõtmisviga ja ebakindlus, mis tähendab, et x-i väärused on täpselt teada (me ei ennusta neid). See eeldus on tähtis siis, kui püüame anda regresionimodelile põhjusliku tõlgenduse. Mudelis $Y = \alpha + \beta_1 X_1 + \beta_2 X_2$ viib mõõtmisviga X_2 -s β_2 koefitsiendi nulli suunas ja ühtlasi vähendab $\beta_2 X_2$ liikme mõju β_1 fittimisel.
6. X-muutuja ei ole konstant vaid sisaldab erinevaid väärusi.
7. Mitme prediktoriga lineaarse regressiooni puhul tuleb sisse veel kollinearuse eeldus: me eeldame, et ükski prediktorite paar pole täiuslikult lineaarselt korreleeritud (pole lineaarne funktsioon üksteisest).

Eeldused praktilise tähtsuse järjekorras:

1. Valiidsus – sa mõõdad asju, mis on relevantsed teadusliku küsimuse seisukoost. Näiteks, kui soovite mõõta kolesterooli alandava ravimi mõju, on

mõistlik mõõta suremust, mitte pelgalt kolesteroli taset veres.

2. Esinduslikkus – andmed peaksid olema esinduslikud laiema populatsiooni suhtes. Väikesed ja kallutatud valimid ei ole sageli esinduslikud.
3. Lineaarsus ja sellest tulenev mudeli additiivsus. Väga tähtis on, et lineaarse regressiooniga mõõdetavad seosed oleks ka tõesti lineaarsed. Kui lineaarsusega on probleeme, võib aidata prediktorite transformeerimine ($\log(x)$ või $1/x$) või uute prediktorite mudelisse lisamine. Samuti on võimalik prediktoriteks samasse mudelisse panna nii x kui x^2 . Näiteks kui me paneme mudelisse nii muutuja *vanus* kui ka *vanus*², saame modelleerida seost, kus y vanuse kasvades alguses kasvab ja siis kahaneb (aga ka U kujulist seost vanusega). Sellisel juhul võib olla ka mõistlik rekodeerida vanus kategooriliseks muutujaks (näit 4 vanuseklassi), mille tasemeid saab siis ükshaaval vaadata.
4. Sõltumatus. Selle eelduse rikkumine viib liiga kitsastele usalduspiiridele.
5. Vigade vordne varieeruvus (homoskedastilisus) ja vigade normaalsus on vähemtähtsad. Log-normaalsete vigadega võiks lineaarsel regresioonil mudeldada $\log(Y)$ skaalas (vähimruutude meetodil) või Bayesi regresioonil mittelineaarsel lognormaalset tõepäramudelit kasutades (vt. ptk 13).
6. Kollineaarsus. Täieliku kolineaarsuse korral mudel ei lahendu, aga sellisel juhul on põhjuseks enamasti viga mudeli spetsifitseerimisel. Osaline, aga ikkagi väga kõrge, kollineaarsus, mis õnnekseks on praktikas pigem haruldane, viib koefitsientide laiadele veapiiridele. Kui veapiirid pole laiad, siis pole ka kollineaarsust.

Regressioon kui kirjeldus ja kui põhjuslik hüpotees

Regressioonanalüüs võib vaadelda 1) empiirilise kirjeldusena y ja x -i koosvarieerumisest või 2) muutujate vaheliste põhjuslike suhete analüüsina. Esimesel juhul ei tõlgenda me x ja y suhet x -i mõjuna y -le. Seega, senikaua kui mudeli fit väljaspool andmeid, mida kasutati selle mudeli fittimiseks, on piisavalt hea, et me fitime vale struktuuriga mudeli. Kui me fitime 2 mudelit (i) $Y = \alpha + \beta_1 X_1$ ja (ii) $Y = \alpha + \beta_1 X_1 + \beta_2 X_2$, siis eeldame, et kahe mudeli β_1 koefitsiendid tulevad erinevad. Aga sellest pole midagi, sest need kirjeldavad mõlemal juhul vaid empiirilisi seoseid.

Teisel, põhjuslikul juhul on kõik teisiti. Eeldades et X_2 on üks Y -i põhjustest, on nütüd esimese mudeli veakomponendis peidus ka $\beta_2 X_2$. Kui X_1 ja X_2 on omavahel korreleeritud, siis tekib meil seetõttu ka korrelatsioon X_1 ja veakomponendi vahel – ja see rikub mudeli eeldusi, kallutades mudeli fittimisel meie hinnangut β_1 -le, mislăbi osa X_2 mõjust Y -le omistatakse ekslikult X_1 -le. See kõik juhtub siis, kui teise mudeli β_2 ei ole null ja esineb X_1 ja X_2 vaheline korrelatsioon.

Selle kallutatuse tõlgendamine sõltub omakorda X_1 ja X_2 vahelise põhjusliku

seose struktuurist. Oluline on mõista, et mudeli enda struktuuris pole vähimatki põhjuslikku infot - mudel ei tea isegi sellise asja nagu põhjuslikkus olemasolust. Seega on meil lisaks regressioonimudelile vaja sellest iseseisvat põhjuslikku mudelit, mille formuleerime puhtalt teaduslikest asjaoludest lähtuvalt.

Kõige lihtsam selline mudel vastab randomiseeritud ja kontrollitud eksperimentile, kus me võrdleme katse ja kontrolltingimus. Siin me usume, et kui katsetingimuse rakendamine (näiteks ravimi manustamine) mõjutab mingis kindlas suunas katse väljundit (näiteks suremust), ja seda võrreldes kontrolltingimusega (näiteks platseeboga), siis me oleme näidanud, et vastav ravim vähendab suremust. Seega on meie põhjuslik skeem ravim → suremus ja regressioonimudel suremus~ravim, mis sisuliselt taandub kahe gruppi keskmiste suremuste võrdlusele.

Kuidas on aga asjalood siis, kui meil ei lubata katset teha? Näiteks, kuidas määrama suitsetamise mõju kopsuvähile? Siin ei ole meil tegemist randomiseeritud katsega (me ei tohi jagada populatsiooni juhuslikult kahte gruppia ja sundida neist ühte suitsetama). Seega peame kasutama statistilisi meetodeid, et kontrollida oma tulemust nn confounderite vastu. Siin on lihtsaim võimalus regressioonimudel vähk ~ suitsetamine + muutuja_1 + ...

Aga muutujaid on maaailmas palju ja meil peab olema mingi reegel, mille järgi otsustada, millised muutujad additiivsesse mudelisse sisse panna ja millised välja jätta. Mudeli ennustusjõu maksimeerimine siin ei aita. Selle asemel peame mõistma võimalike põhjuslike skeemide suhet mitmese regressioonimudelitega. Põhjuslikud skeemid on nagu legod, mis koosnevad järgmistest põhiosistest e hitusplokkidest.

1. toru: $x \rightarrow z \rightarrow y$
2. kahvel: $x <- z \rightarrow y$
3. laupkokkupõrge: $x \rightarrow z <- y$
4. järglane: see on toru, kus z -i juurest hargneb veel üks nool A -le. Siin saame me mudelisse A lisades ligikaudu sama tulemuse, mis z -i lisades. Seega, kui z -i väärtsused pole meile teada, võime hädaga ka A -d kasutada.

Kui me tahame teada, kas x mõjutab y -t, siis toru puhul mudel $y \sim x + z$ vähendab x -i mõju (sest see mõju käib läbi z -i). Samas, mudel $y \sim x$ näitab x -i mõju. Seega, kumba mudelit kasutada sõltub sellest, kas me tahame näidata x -i otsest või kaudset mõju y -ile.

Kahvli puhul regressioonimudel $y \sim x$ näitab x -i mõju y -le (ehkki meie põhjuslikkuse mudelis puudub x ja y vaheline põhjuslik seos), aga mudel $y \sim x + z$ välistab selle mõju. Seega peaksime sellise põhjusliku hüpoteesi korral mudelisse z -i sisse panema, sest see aitab kontrollida z konfounding mõju vastu.

Laupkokkupõrke korral on olukord eelnevaga vastupidine. Nüüd avab mudel $y \sim x + z$ tagaukse ja laseb z -i segava mõju mudelisse sisse, mis tekitab meile võlts-põhjusliku suhte x ja y vahel.

Näiteks võib meil tekkid olukord, kus testime suitsetamise mõju vähile, aga me usume, et inimeste vanus mõjutab iseseisvalt nii vähki kui suitsetamist (vanemad inimesed surevad rohkem, aga nad ka suitsetavad rohkem). Selles põhjuslikus skeemis töötab vanus kahvlina, millega arvestamiseks tuleb see regressioonimudelisse muutujana sisse panna: vähk ~ suitsetamine + vanus.

Chapter 5

Kaks lineaarse mudeli laiendust

```
library(tidyverse)
library(scatterplot3d)
library(viridis)
library(ggeffects)
library(broom)
library(car)
```

5.1 Mitme sõltumatu prediktoriga mudel

Esiteks vaatame mudelit, kus on mitu prediktorit x_1, x_2, \dots, x_n , mis on aditiivse mõjuga. See tähendab, et me liidame nende mõjud, mis omakorda tähendab, et me usume, et $x_1 \dots x_n$ mõjud y -i värtusele on üksteisest sõltumatud. Mudel on siis kujul

$$y = a + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

Mitme prediktoriga mudeli iga prediktori tõus (beta koefitsient) ütleb, mitme ühiku võrra ennustab mudel y muutumist juhul kui see prediktor muutub ühe ühiku võrra ja kõik teised prediktorid ei muutu üldse (Yule, 1899).

Milliseid muutujaid (regressoreid) peaks üks hea lineaarne mudel sisaldama, milliseid peaks me mudelist välja viskama ja milliseid igal juhul sisse panema? Matemaatiliselt põhjustab regressorite eemaldamine ülejäändud regressorite koefitsientide ebakonsistsust, välja arvatud siis, kui (i) välja visatud regressorid

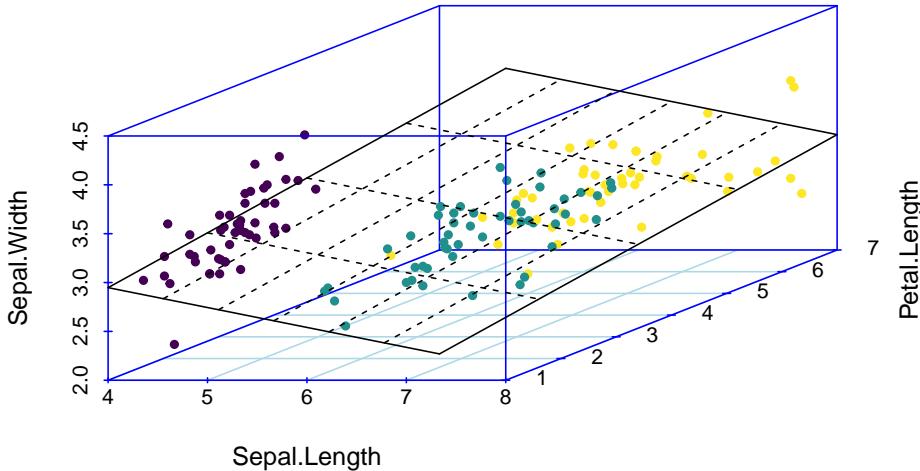


Figure 5.1: Regressioonitasand 3D andmetele. Kahe prediktoriga mudel, kus Sepal.Length ja Petal.Length on prediktorid ja Sepal.Width ennustatav muutuja.

ei ole korreleeritud sisse jäetud regressoritega või (ii) välja vistatud regressorite koefitsiendid võrduvad nulliga, mis muudab nad ebarelevantseteks. Kuidas sa tead, et kõik vajalikud regressorid on sul üldse olemas (olematuid andmeid ei saa ka mudelisse lisada)? Loomulikult ei teagi, mis tähendab lihtsalt, et mudeldamine on keeruline protsess, nagu teaduski. Pane ka tähele, et koefitsiendi “mitte-oluline” p väärthus ei tähenda iseenesest, et koefitsient töenäoliselt võrdub nulliga või on nulli lähedal, vaid seda, et meil pole piisavalt andmeid, et vastupidist kinnitada. Koefitsiendi hinnangu usalduspiirid on selles osas palju parem töövahend.

Kui meie andmed on kolmedimensionaalsed (me mõõdame igal mõõteobjektil kolme muutujat) ja me tahame ennnustada ühe muutuja väärustust kahe teise muutuja väärustute põhjal (meil on kaks prediktorit), siis tuleb meie kolme parameetriga lineaarne regressioonimudel tasapinna kujul. Kui meil on kolme prediktoriga mudel, siis me liigume juba neljamõõtmelisse ruumi.

Seda mudelite saab kaeda 2D ruumis, kui kollapseerida kolmas mõõde konstandile.

```
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width, color = Species)) +
  geom_point() +
  xlim(4, 8) +
  scale_color_viridis(discrete = TRUE) +
  theme(title = element_text(size = 8))
p1 <- p + geom_abline(intercept = coef(m2)[1], slope = coef(m2)[2]) +
  labs(title = deparse(formula(m2)))
m1 <- lm(Sepal.Width ~ Sepal.Length, data = iris)
p2 <- p + geom_abline(intercept = coef(m1)[1], slope = coef(m1)[2]) +
```

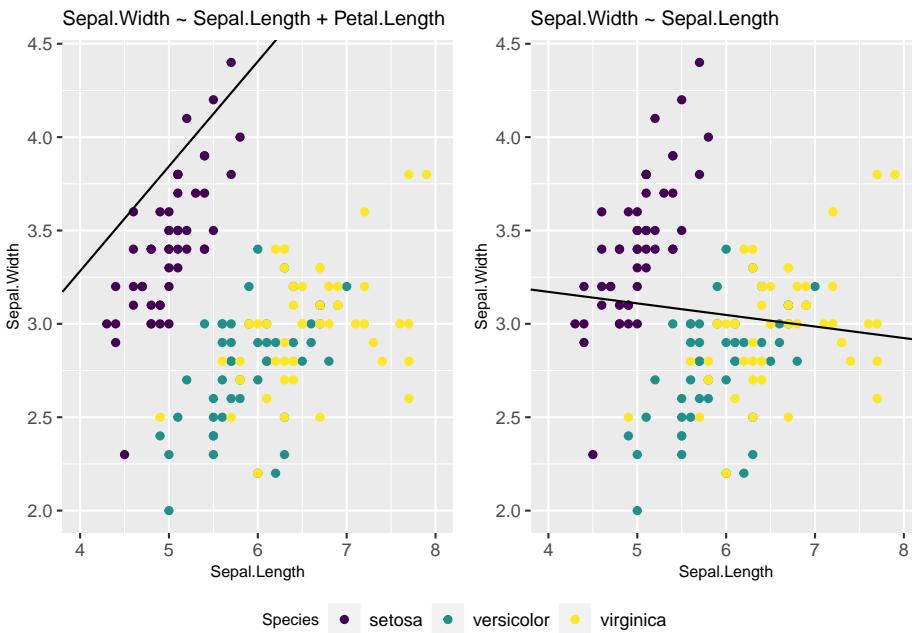


Figure 5.2: 2D-le kollapseeritud graafiline kujutus 3D andmete põhjal fititud mudelist. Vasemal, muutuja Petal.Length on kollapseeritud konstandile. Siin on regressioonjoon hoopis teises kohas, kui lihtsas ühe prediktoriga mudelis (paremal).

```
  labs(title = deparse(formula(m1)))
devtools::source_gist("8b4d6ab6a333ef1cd14e8067c3badbae", filename = "gridArrangeSharedLegend.R")
gridArrangeSharedLegend(p1, p2)
```

Võrreldes mudelite m1 (üks prediktor) ja m2 (kaks prediktorit) Sepal.Length (b_1) koefitsienti on näha, et need erinevad oluliselt.

```
coef(m1)
#> (Intercept) Sepal.Length
#>      3.4189     -0.0619
coef(m2)
#> (Intercept) Sepal.Length Petal.Length
#>      1.038       0.561      -0.335
```

Kumb mudel on siis parem? AIC-i järgi on m2 kõvasti parem kui m1, lisakoeffi-sendi (Petal.Length) kaasamisel mudelisse paranes oluliselt selle ennustusvõime.

```
AIC(m1, m2)
#>   df   AIC
```

```
#> m1  3 179.5
#> m2  4  92.1
```

Ennustused sõltumatute prediktoritega mudelist

Siin on idee kasutada fititud mudeli struktuuri ennustamaks y keskmisi väärtsusi erinevatel x_1 ja x_2 väärustel. Kuna mudel on fititud, on parameetrite väärtsused fikseeritud.

```
## New sepal length values
Sepal_length <- seq(min(iris$Sepal.Length), max(iris$Sepal.Length), length.out = 10)
## Keep new petal length constant
Petal_length <- mean(iris$Petal.Length)
## Extract model coefficients
a <- coef(m2)["(Intercept)"]
b1 <- coef(m2)["Sepal.Length"]
b2 <- coef(m2)["Petal.Length"]
## Predict new sepal width values
Sepal_width_predicted <- a + b1 * Sepal_length + b2 * Petal_length

plot(Sepal_width_predicted ~ Sepal_length, type = "b", ylim = c(0, 5), col = "red")
# Prediction from the single predictor model
abline(m1, lty = "dashed")
```

Nüüd joonistame 3D pildi olukorras, kus nii x_1 kui x_2 omandavad rea väärtsusi. Mudeli ennustus on ikkagi sirge kujul – mis sest, et 3D ruumis.

```
#> Warning: `data_frame()` is deprecated, use `tibble()``.
#> This warning is displayed once per session.
```

5.2 Interaktsioonimudel

Interaktsioonimudelis sõltub ühe prediktori mõju teise prediktori väärusest:

$$y = a + b_1 x_1 + b_2 x_2 + b_3 x_1 x_2$$

Ekvivalentne viis interaktsiooni spetsifitseerida on läbi võrrandisüsteemi:

$$y = a + \gamma x_1 + b_2 x_2$$

$$\gamma = b_1 + b_3 x_2$$

Siit on hästi näha, et me teeme kaks lineaarset regressiooni, milles teine modelleerib x_1 muutuja koefitsiendi sõltuvust x_2 muutuja väärusest.

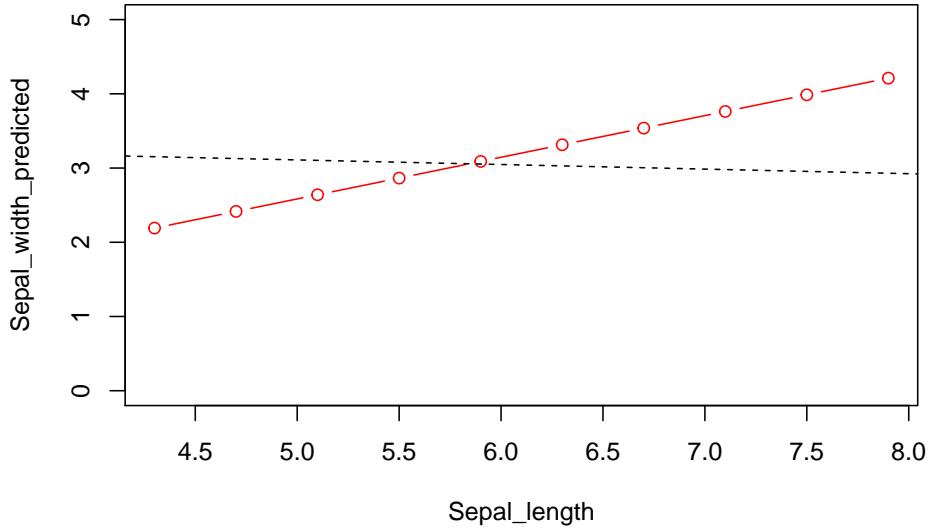


Figure 5.3: Ennustatud y väärustused erinevatel x_1 väärustel kui x_2 on konstantne, punane joon. Katkendjoon, ühe prediktoriga mudeli ennustus.

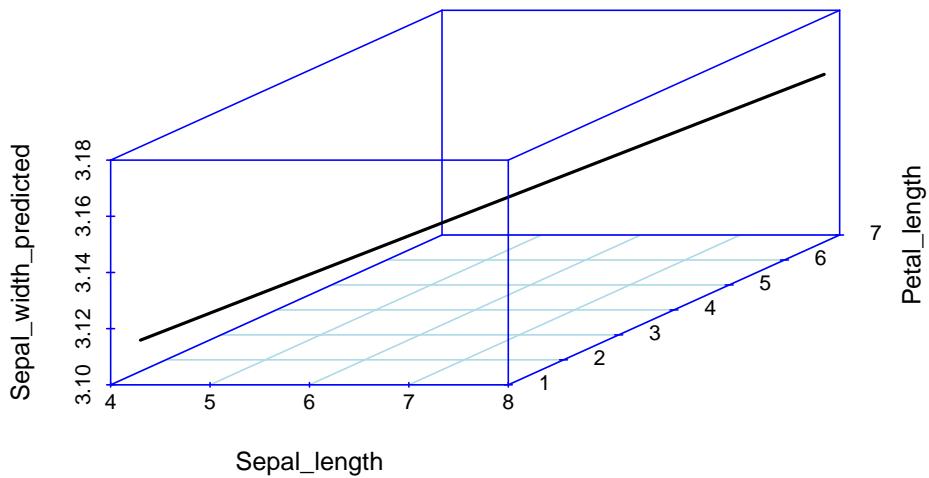


Figure 5.4: Kahe prediktoriga mudeli ennustus 3D ruumis.

Samamoodi kehtib ka ümberkirjutus

$$y = a + \gamma x_2 + b_1 x_1$$

$$\gamma = b_2 + b_3 x_1$$

mis tähendab, et ühtlasi modelleerime me ka x_2 koefitsiendi sõltuvust x_1 -st. Mudeli koefitsientide tõlgendamise teeb keeruliseks, et gamma tõlgendamisel tuleb arvesse võtta kolm asja – b_2 , b_3 ja x_1 .

Kaks muutujat võivad intakteeruda sõltumata sellest, kas nad on korreleeritud või mitte – interaktsioon ei impitseeri korrelatsiooni, ega vastupidi.

Sageli on nii, et prediktoreid, mille mõju y -le on suur, tasub mudeldada ka interaktsionimudelis (näiteks suitsetamise mõju vähimudelites kipub olema interaktsioniga). Interaktsionimudelis on b_1 koefitsient otse tõlgendatav ainult siis, kui $x_2 = 0$ (ja b_2 ainult siis, kui $x_1 = 0$).

Kui interaktsionimudel fititakse tsentreeritud x -muutujate peal, mille keskväärtus = 0 (või standardiseeritud muutujatel), siis muutub koefitsientide tõlgendamine lihtsamaks:

- b_1 annab y tõusu, kui x_1 tõuseb 1 ühiku võrra ja x_2 on fikseeritud oma keskväärtusel
- b_2 annab y tõusu, kui x_2 tõuseb 1 ühiku võrra ja x_1 on fikseeritud oma keskväärtusel).
- b_3 ütleb, kui palju muutub x_1 mõju y -le, kui x_2 muutub ühe ühiku võrra. Samamoodi, b_3 ütleb, kui palju muutub x_2 mõju y -le, kui x_1 muutub ühe ühiku võrra.

NB! Ärge standardiseerige faktormuutujaid ehk *dummy*-regressoreid kujul 1, 0 – neid on lihtsam tõlgendada algsel kujul 0/1 skaalas.

Edaspidi õpime selliseid mudeleid graafiliselt tõlgendama, kuna koefitsientide otse tõlgendamine ei ole siin sageli perspektiivikas.

Interaktsionimudelis sõltub x_1 mõju tugevus y -le x_2 väärustusest.

Selle sõltuvuse määra kirjeldab b_3 (x_1 ja x_2 interaktsiooni tugevus).

Samamoodi ja sümmeetriliselt erineb ka x_2 mõju erinevatel x_1 väärustel. Ainult siis, kui $x_2 = 0$, ennustab x_1 tõus 1 ühiku võrra y muutust b_1 ühiku võrra.

Kui meil on mudelis interaktsioniliige $x_1 x_2$, siis on enamasti mõistlik ka lisada eraldi liikmetena ka x_1 ja x_2 .

Näiteks mudel, milles on pidev y -muutuja, pidev prediktor “education” ja binarne prediktor “sex_male” (1 ja 0):

$$score = a + b_1 * education + b_2 * sex_{male} + b_3 * education * sex_{male}$$

Variandis

$$score = a + b_1 * education + b_3 * education * sex_{male}$$

surume meeste ja naiste intercepti pidevale muutujale “education” ühte punkti, aga samas modelleerime sellele erinevad tõusud meeste ja naiste lõikes.

Samamoodi, variandis

$$score = a + b_2 * sex_{male} + b_3 * education * sex_{male}$$

on naiste tõus surutud nulli, aga interceptid võivad erineda, mis on kokkuvõttes üsna imelik, kuigi tehniliselt on mudel ok ja seda võib edukalt fittida.

Ja variandis

$$score = 0 + b_3 * education * sex_{male}$$

On meil meeste ja naiste intercept surutud nulli, aga meeste ja naiste tõusud võivad erineda.

Kui meil on kaks faktor-prediktorit, siis mudel kujul

$$y = 0 + b_3 x_1 x_2$$

Mudeldab eraldi nende faktorite tasemete kõikvõimalud kombinatsioonid.

Oletame, et meil on lisaks pidevale prediktorile x_1 ka faktor-prediktor x_2 . Diskreetsed e faktor-prediktorid rekodeeritakse automaatselt nn *dummy*-muutujateks. Kaheivalentse e binaarse muutuja, näit $sex = c(\text{"male"}, \text{"female"})$, korral läheb regressioonivõrrandisse uus dummy-muutuja, sex_female , kus kõik emased on 1-d ja isased 0-d. Üldine intercept vastab siis isaste mõjule ja sex_female intercept annab emaste erinevuse isastest. Kui meil on n-tasemega diskreetne muutuja, rekodeerime selle n-1 *dummy*-muutujana, milles igaüks on 0/1 kodeeringus ja milles igaühe interceptid annavad erinevuse null-taseme (selle taseme, mis ei ole rekodeeritud *dummy*-muutujana) interceptist. Mudeli seisukohast pole oluline, millise faktortunnuse taseme me nulltasemeks võtame. Terminoloogiliselt on meie n-tasemega faktortunnus *seletav muutuja (explanatory variable)*, milles tehakse n-1 *regressorit*. Seega tehniliselt on mudeli liikmed regressorid, mitte seletavad muutujad. Üks seletav muutuja võib anda välja mitu regressorit (nagu eelmises näites) ja üks regressor võib põhineda mitmel muutujal (näit x_1x_2 interaktsioniterm).

Interaktsioonimudeli 2D avaldus on kurvatuuriga tasapind, kusjuures kurvatuuri määrab b_3 .

Interaktsiooniga mudel on AIC-i järgi pisut vähem eelistatud võrreldes kahe prediktoriga mudeliga m_2 . Seega, eriti lihtsuse huvides, eelistame m_2 -e.

```
m3 <- lm(Sepal.Width ~ Sepal.Length + Petal.Length + Sepal.Length * Petal.Length, data = iris)
AIC(m1, m2, m3)
#>      df    AIC
#> m1   3  179.5
#> m2   4  92.1
#> m3   5  93.4
```

Ennustused interaktsioonimodelist

Kõigepealt anname rea väärtsusi x_1 -le ja hoiame x_2 konstantsena.

```
Petal_length <- mean(iris$Petal.Length)
a <- coef(m3)["(Intercept)"]
b1 <- coef(m3)["Sepal.Length"]
b2 <- coef(m3)["Petal.Length"]
b3 <- coef(m3)["Sepal.Length:Petal.Length"]
Sepal_width_predicted <- a + b1 * Sepal_length + b2 * Petal_length + b3 * Sepal_length
plot(Sepal_width_predicted ~ Sepal_length, type = "l", ylim = c(2, 6))
abline(coef(m2)[c("(Intercept)", "Sepal.Length")], lty = "dashed")
```

Nagu näha viib korrutamistehe selleni, et interaktsioonimudeli tõus erineb ilma interaktsionita mudeli tõusust.

Kui aga interaktsioonimudel plottida välja 3D-s üle paljude x_1 ja x_2 väärustele, saame me regressioonikurvi (mitte sirge), kus b_3 annab kurvatuuri.

Vau! See on alles ennustus!

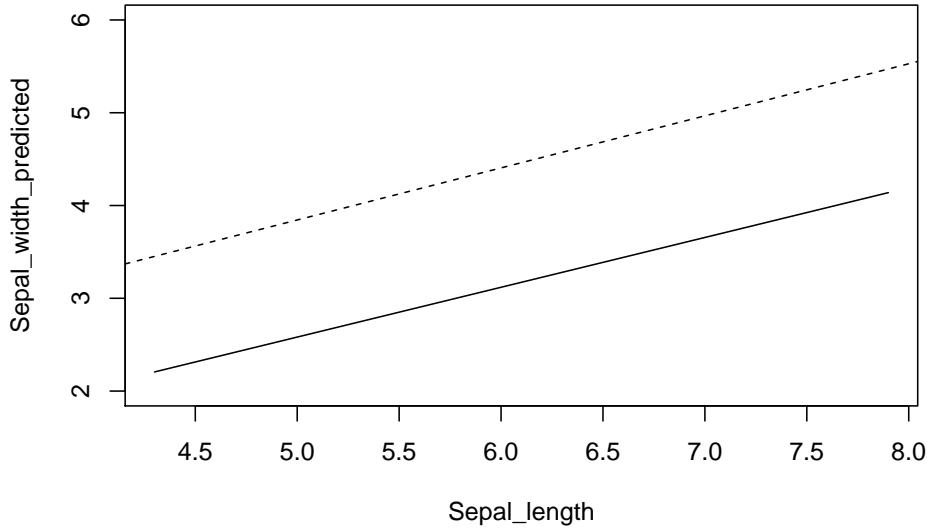


Figure 5.5: Ennustus interaktsioonimudelist, kus x_1 (Sepal_Length) on antud rida väärtsusi ja x_2 (Petal_length) hoitakse konstantsena (pidevjoon). Interaktsioonimudeli regressioonijoon on paraleelne ilma interaktsioonita mudeli ennustusele (katkendjoon).

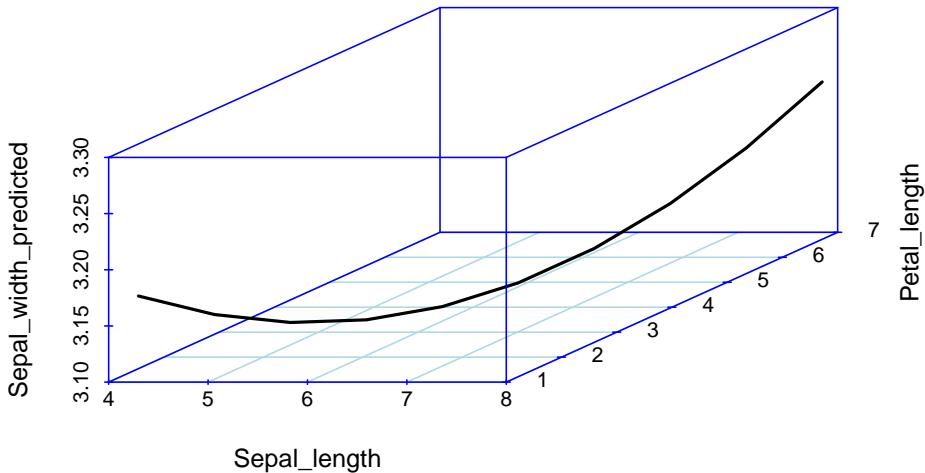


Figure 5.6: Ennustused 3D interaktsioonimudelist üle paljude x_1 (Sepal_Length) ja x_2 (Petal_length) väärustele.

Chapter 6

Vähimruutude meetodiga fititud mudelite töövoog – `lm()`

Kuna `lm()` funktsiooniga ja bayesi meetodil fititud mudeliobjektidega töötamine on mõnevõrra erinev, öpetame seda eraldi. Siinkohal anname põhilise töövoo `lm()` mudelobjektide inspekteerimiseks.

Töötame `m3` mudeliobjektiga, mis on interaktsionimudel:

`Sepal.Width ~ Sepal.Length * Species`
ehk

$$Sepal.Width = a + b_1 * Sepal.Length + b_2 * Species + b_3 * Sepal.Length * Species$$

```
library(ggeffects)
m3 <- lm(Sepal.Width ~ Sepal.Length * Species, data = iris)
```

1. vaatame mudeli koefitsiente

```
tidy(m3)
#> # A tibble: 6 x 5
#>   term            estimate std.error statistic  p.value
#>   <chr>           <dbl>     <dbl>      <dbl>    <dbl>
#> 1 (Intercept) -0.569      0.554     -1.03  3.06e- 1
```

```
#> 2 Sepal.Length          0.799    0.110    7.23 2.55e-11
#> 3 Speciesversicolor   1.44     0.713    2.02 4.51e- 2
#> 4 Speciesvirginica   2.02     0.686    2.94 3.85e- 3
#> 5 Sepal.Length:Speciesversicolor -0.479   0.134    -3.58 4.65e- 4
#> 6 Sepal.Length:Speciesvirginica -0.567   0.126    -4.49 1.45e- 5
```

Interaktsionimudeli koefitsientide jöllitamine on sageli tühi töö ja vaimu närimine. Õnneks on meil muid meeotodeid, kuidas lm() mudelitega töötada.

Võrdluseks - nii fitime eraldi mudeli igale irise liigile. Tulemus on tegelikult identne interaktsionimudeliga kategoorilisele muutujale (Species), aga koefitsendid on otse tõlgendatavad. Samas, interaktsionimudelit saab fittida ka pidevate muutujate!

```
iris %>% split(. $Species) %>%
  map(~ lm(Sepal.Width ~ Sepal.Length, data = .)) %>%
  map(summary) %>%
  map_dfr(~ broom::tidy(.), .id = "Species")
#> # A tibble: 6 x 6
#>   Species   term      estimate std.error statistic p.value
#>   <chr>     <chr>      <dbl>     <dbl>      <dbl>    <dbl>
#> 1 setosa   (Intercept) -0.569     0.522     -1.09 2.81e- 1
#> 2 setosa   Sepal.Length  0.799     0.104      7.68 6.71e-10
#> 3 versicolor (Intercept) 0.872     0.445      1.96 5.56e- 2
#> 4 versicolor Sepal.Length  0.320     0.0746     4.28 8.77e- 5
#> 5 virginica (Intercept)  1.45      0.431      3.36 1.55e- 3
#> 6 virginica Sepal.Length  0.232     0.0651     3.56 8.43e- 4
```

Adjusteeritud r2 tasub eraldi üle vaadata.

```
summary(m3)$adj.r.squared
#> [1] 0.61
```

0.61 tähendab, et mudel suudab seletada mitte rohkem kui 61% y-muutuja (Sepal.Width) varieeruvusest.

2. Testime mudeli eeldusi

Nii saab fititud väärtsused (.fitted), residuaalid (.resid), fittitud väätustest standardvead (.se.fit). Residuaal = y data value - fitted value. Seega positiivne residuaal näitab, et mudeli ennustus keskmisele y väärtsusele mingil x-muutujate väärtsusel on madalam kui juhutb olema tegelik y-i andmepunkti väärtsus. See võib olla tingitud y-muutuja normaalsest bioloogilisest varieeruvusest, aga ka sellest, et mudel ei kirjelda täiuslikult x-ide ja y tegelikku seost.

```
(a_m3 <- augment(m3))
#> # A tibble: 150 x 10
#>   Sepal.Width Sepal.Length Species .fitted .se.fit  .resid  .hat .sigma
#>   <dbl>       <dbl> <fct>     <dbl>   <dbl>    <dbl>   <dbl>   <dbl>
#> 1     3.5       5.1 setosa     3.50  0.0399 -0.00306 0.0215  0.273
#> 2     3          4.9 setosa     3.34  0.0403 -0.343   0.0218  0.272
#> 3     3.2       4.7 setosa     3.18  0.0512  0.0163  0.0354  0.273
#> 4     3.1       4.6 setosa     3.10  0.0591 -0.00380 0.0471  0.273
#> 5     3.6       5          setosa     3.42  0.0385  0.177   0.0200  0.273
#> 6     3.9       5.4 setosa     3.74  0.0581  0.157   0.0455  0.273
#> # ... with 144 more rows, and 2 more variables: .cooks <dbl>,
#> #   .std.resid <dbl>
```

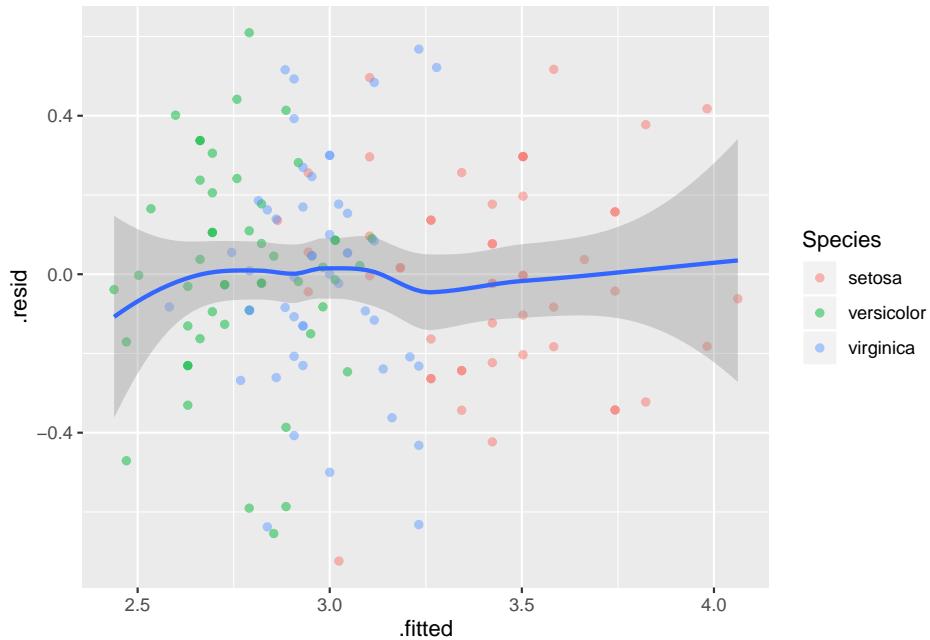
.hat >1 sugereerib high leverage andmepunkte

.std.resid on studentiseeritud residuaal, mis on sd ühikutes (.resid/sd(.resid))

Lineaarsus - residuaalid~fitted plot

Residuals vs fitted plot testib lineaarsuse eeldust - kui .resid punktid jaotuvad ühtlaselt nulli ümber, siis mudel püüab kinni kogu süstemaatilise varieeruvuse teie andmetest ja see mis üle jääb on juhuslik varieeruvus.

```
ggplot(a_m3, aes(`.fitted`, `.resid`)) +
  geom_point(aes(color=Species), alpha=0.5) +
  geom_smooth()
```



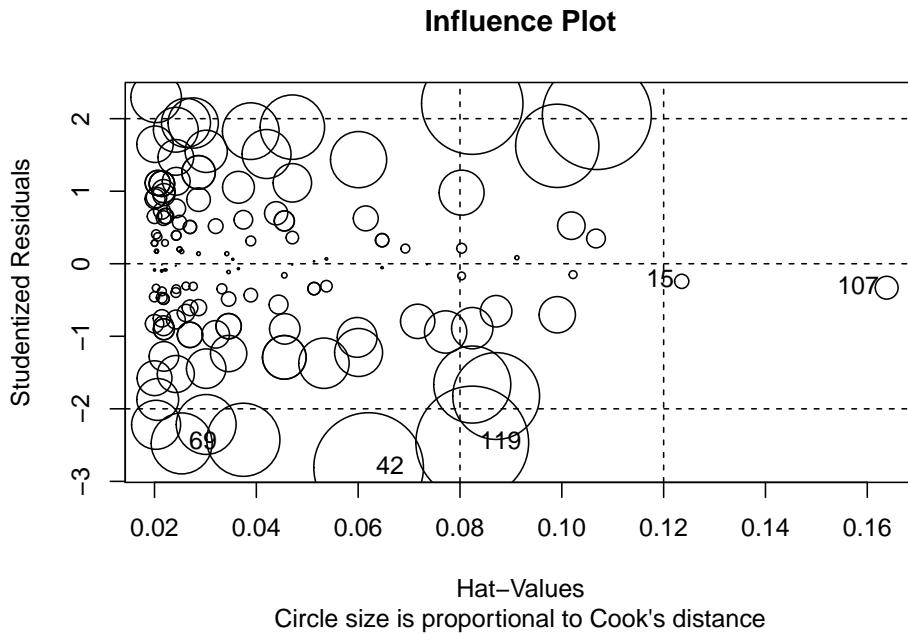
Mõjukuse plot

- *outlierid* – studentideeritud residuaalid > 2 või < -2 . Studentiseeritud residuaali saab (ligikaudu) jagades vaatluse residuaali residuaalide standardhälbgaga. See protseduur võimaldab paremini võrrelda erinevate vaatluste residuaale.

Standardiseeritud residuaali arvutamine: Kui E_i on i-s residuaal, k on mudeli regressorite arv ja n on vaatluste arv, siis $h_i = 1/n + E_i / \sum E^2$, $S_E = (E^2 / (n - k - 1))^{1/2}$ ja $E_{st} = E_i / (S_E(1 - h_i)^{1/2})$ kus E_{st} on standardiseeritud residuaal, mis suurtel valimitel on väga sarnane studentiseeritud residuaaliga (mis erineb selle poolest, et välistab iga residuaali $S \sim E_{st}$ -st seda residuaali genereerinud vaatluse). Kui n on suur, siis kehtib enam-vähem seos $E_{st} = E_i / sd(E)$, kus E_{st} on nii standardiseeritud kui studentiseeritud residuaal.

- *high leverage* vaatlused – $\hat{h}_{ij} > 1$ - sugereerib *high leverage* vaatlust. Keskmne \hat{h}_{ij} value $= (k + 1)/n$, kus k on regressorite arv (mitte arvestades intercepti) ja n on vaatluste arv. NB! Kuna *high leverage* vaatlused tömbavad regressioonijoon enda suunas, siis on neil sageli madalad residuaalid (erinevalt outlieritest, mis ei ole *high leverage* vaatlused)

```
library(car)
influencePlot(m3, id.method="identify", main="Influence Plot",
              sub="Circle size is proportional to Cook's distance")
```



```
#>      StudRes   Hat   CookD
#> 15    -0.243 0.1236 0.00139
#> 42    -2.810 0.0621 0.08307
#> 69    -2.477 0.0253 0.02567
#> 107   -0.331 0.1638 0.00359
#> 119   -2.464 0.0824 0.08782
```

Horisontaalsed referentsjooned näitavad 0, 2 ja -2 studentiseeritud residuaale. Vertikaalsed referentsjooned näitavad hat-väärtusi 2h ja 3h.

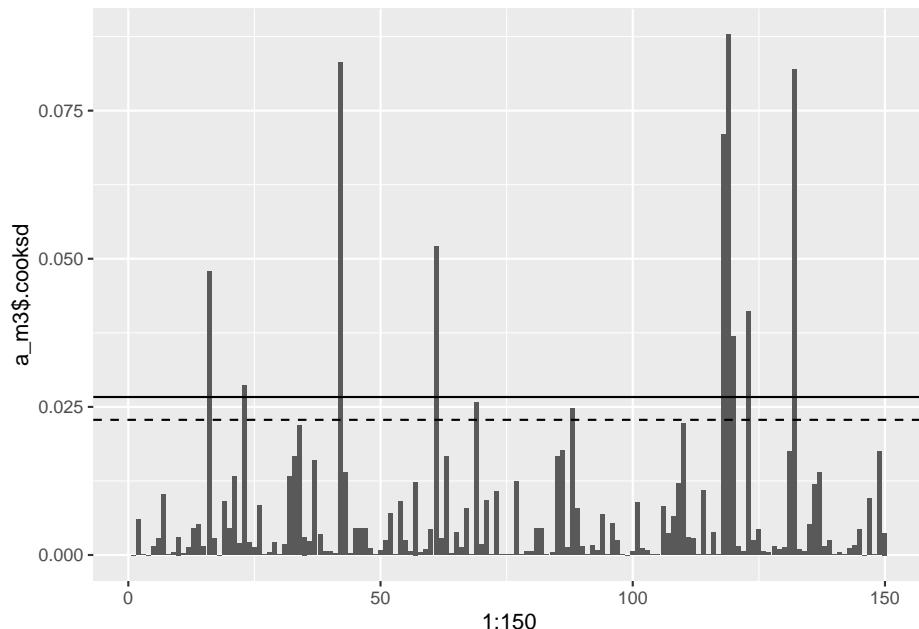
Regressiooni *outlier* on vaatlus, mille y-muutja väärtus on ebatavaline X-muujutuja väärtuse kontekstis. Seega annab *outlier* mudeli fittimisel kõrge residuaaliga punkti. Lihtsalt (mitte-konditsionaalselt) ebatavalised Y-i või X-i väärtused ei pruugi olla *outlierid*. Kui peaks juhum, et *outlier* langeb kokku ebatavalise X-i väärtusega, siis selle punkti eemaldamine muudab märkimisväärtselt mudeli koefitsiente. Selline *outlier* on ühtlasi ka *high leverage* vaatlus. Siit jõuame mõjukate vaatluste (*Influential observations*) defineerimisele — Mõjukus mudeli koefitsientidele = *Leverage* x “*outlierness*”. *High leverage* andmepunktid on x-muutujate ekstreemsed punktid, mille lähedal ei ole n-mõõtmelises ruumis (kui teil on n x-muutujat) teisi punkte. Seetõttu läheb fititud mudel just nende punktide lähedalt mõõda. Mõjukad punktid on tüüpiliselt ka *high leverage* punktid, kuid vastupidine ei kehti!

Cooki kaugus - mõjukus

.cooksd on Cook-i kaugus, mis näitab mõjukust. Rusikareeglinä tähindab cooksd > 3 cooksd keskväärtust, et tegu võiks olla mõjuka vaatlusega. Teine võimalus on pidada mõjukaks igat punkti, mis on kõrgem kui $4/n$. Kolmanadad arvavad jälle, et $.cooksd > 1$ v $.cooksd > 0.5$ viitab mõjukale vaatlusele. Üldiselt on kõigi mudeli eelduste kontrollidega nii, et vastava statistiku jaotuse jäollitamine on sageli kasulikum kui automaatselt mingi *cut-off* järgi talitamine.

Cooki D andmepunktile saame valemist $D_i = \frac{E'_i}{k+1} + \frac{h_i}{1-h_i}$, kus D_i on i-ndale vaatlusele vastav Cooki kaugus ja E'_i on sellele vaatlusele vastav studentiseeritud residuaal.

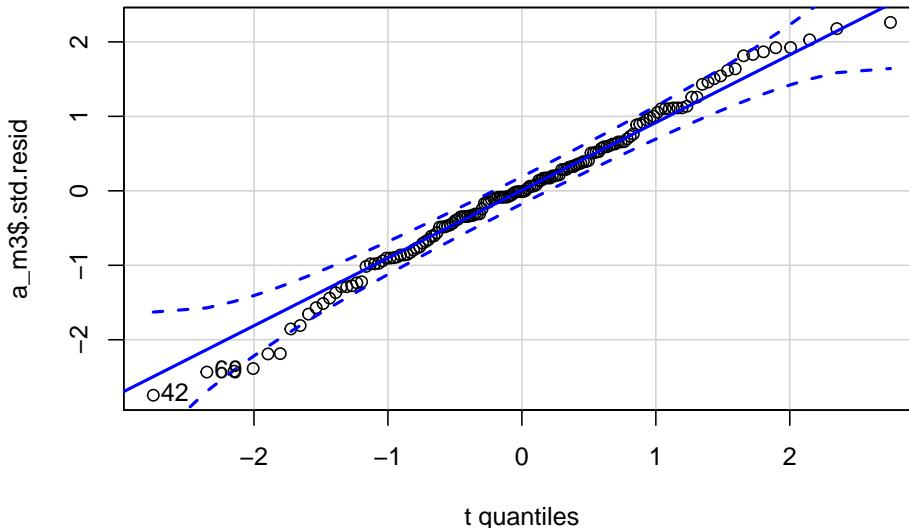
```
ggplot(data = NULL, aes(x = 1:150, y = a_m3$.cooksd)) + geom_col() +
  geom_hline(yintercept = 4/150) +
  geom_hline(yintercept = 3*mean(a_m3$.cooksd), lty = 2)
```



Residuaalide normaalsus - qq plot

Kas residuaalid on normaaljaotusega? NB! studentiseeritud residuaalid on studenti t jaotusega ja üldiselt on targem vaadata neid, kui tavalisi residuaale. Studenti t jaotusele pean ette andma ka vabadusastmete arvu e df-i.

```
car::qqPlot(a_m3$.std.resid, distribution = "t", df=149)
```

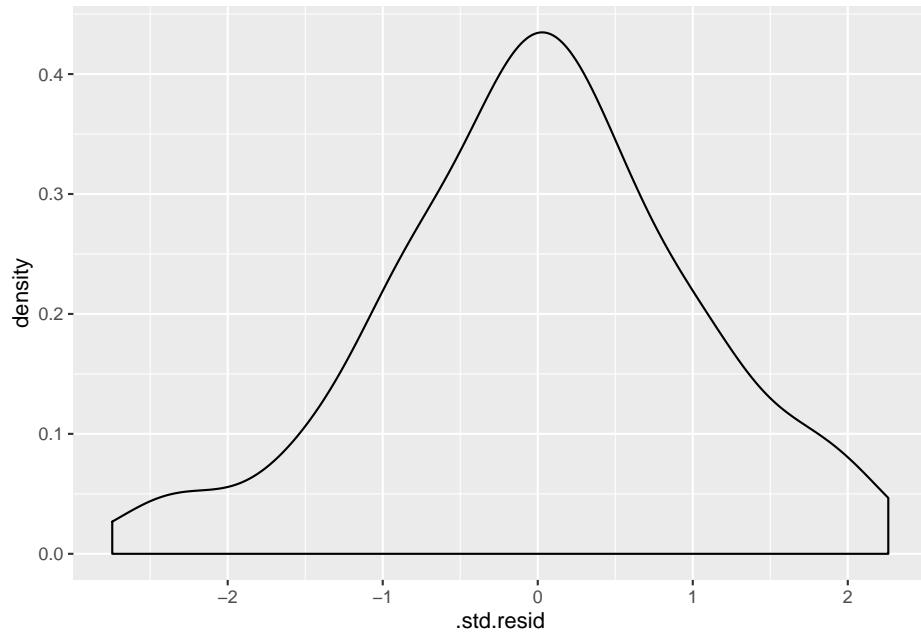


```
#> [1] 42 69
```

QQ-plot näitab erinevust normaaljaotusest (t jaotusest) eelkõige residuaalide jaotuse sabades. Antud juhul on kõik hästi.

Isegi olulisem on vaadata, et residuaalide jaotus ei oleks mitmetipuline. Kui on, siis võib see olla märgiks, et mudelist on piudu mõni faktormuutuja, mis andmetes olevad diskreetsed loomulikud alampopulatsioonid lahku ajaks.

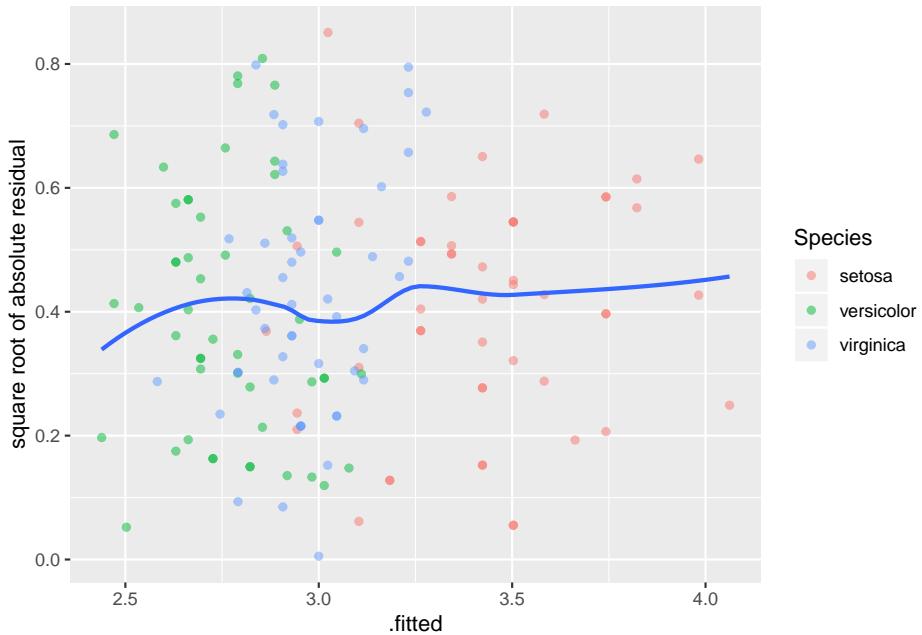
```
ggplot(a_m3, aes(`.std.resid`)) + geom_density()
```



Homoskedastilisus - Scale-location plot

Scale-location plot - homoskedastilisuse eeldust ehk seda, et varieeruvus ei sõltuks prediktormuutuja väärustusest. Y-teljel on ruutjuur studentiseeritud residuaalide absoluutväärustusest

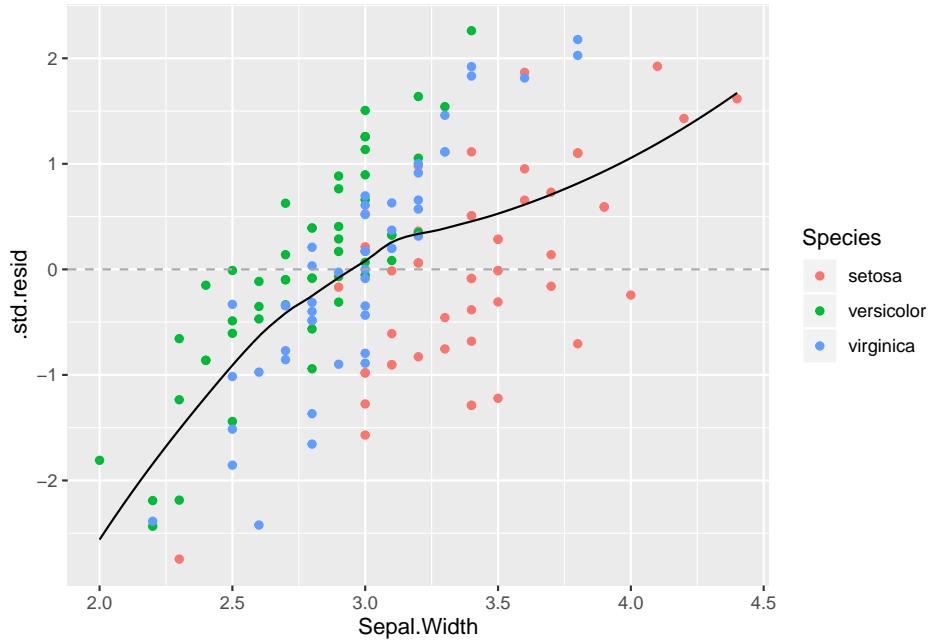
```
ggplot(a_m3, aes(`.fitted`, `resid` %>% abs %>% sqrt)) +
  geom_point(aes(color=Species), alpha=0.5) +
  ylab("square root of absolute residual")+
  geom_smooth(se = FALSE)
```



Residuaalid y ja x muutujate vastu

Kõigepealt residuaalid y-muutja vastu

```
ggplot(a_m3, aes(Sepal.Width, `^std.resid`)) + geom_point(aes(color=Species)) + geom_hline(yintercept = 0)
  geom_smooth( se=F, color="black", size=0.5)
```

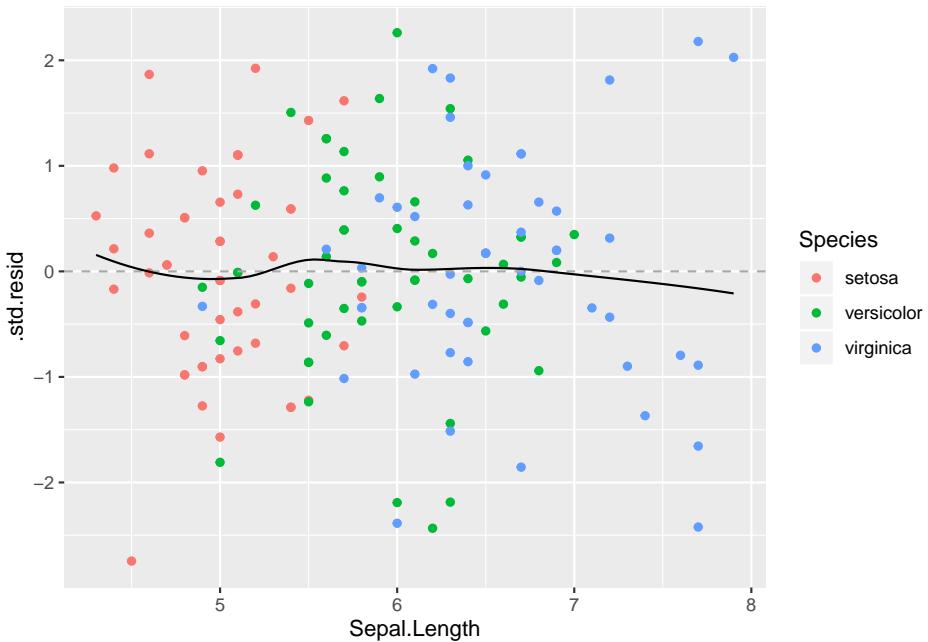


Mudel paistab süstemaatiliselt alahindama Sepal Width-i seal kus Sepal Length on kõrge, ja vastupidi. Horisontaalne punktiirjoon näitab, kus mudel vastab täpselt andmetele.

Studentiseeritud residuaalid sd ühikutes

Ja nüüd residuaalid x-muutuja vastu.

```
ggplot(a_m3, aes(Sepal.Length, `^.std.resid`, color=Species)) +
  geom_point() +
  geom_hline(yintercept = 0, lty =2, color ="darkgrey")+
  geom_smooth(se=F, color="black", size=0.5)
```



Ideaalsed residuaalid!

3. Teeme mudeli põhjal ennustusi (marginal plots)

Me ennustame y-i keskmisi väärtsuseid etteantud x-i väärustel.

`ggpredict()` ennustab y-muutuja väärtsusi ühe x-muutuja vääruste järgi, hoides kõiki teisi x-muutujaid konstantsena.

Kõigepealt võrdleme lihtsa 1 prediktoriga mudeli ennustust kahe prediktoriga mudeli ennustusega

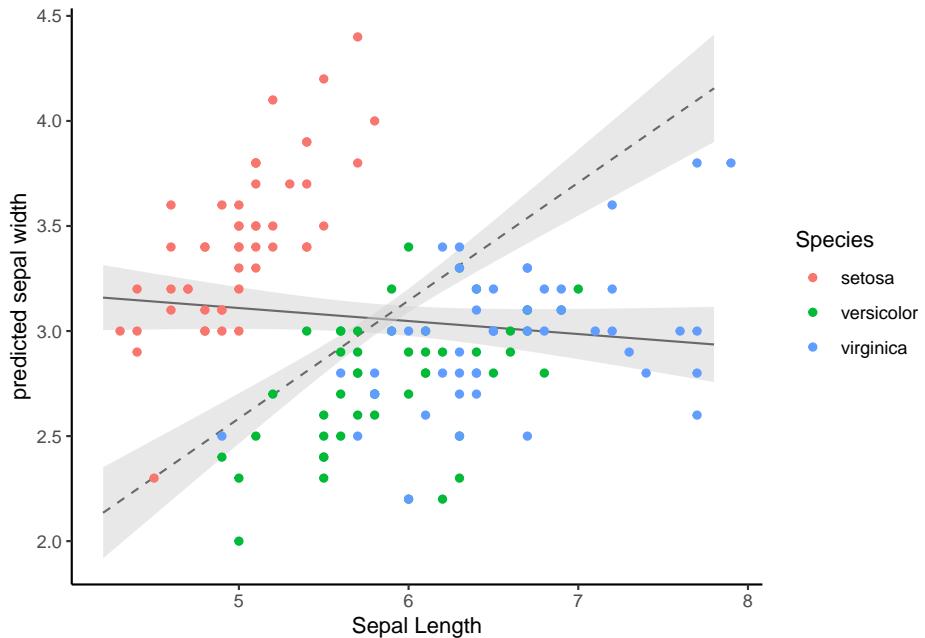
```
lm1 <- lm(Sepal.Width ~ Sepal.Length, data = iris)
lm2 <- lm(Sepal.Width ~ Sepal.Length + Petal.Length, data = iris)

mydf <- ggpredict(lm1, terms = "Sepal.Length")
mydf2 <- ggpredict(lm2, terms = "Sepal.Length")

ggplot(mydf, aes(x, predicted)) +
  geom_line() +
  geom_ribbon(data = mydf, aes(ymin = conf.low, ymax = conf.high),
              alpha = 0.5, fill="lightgrey") +
  geom_line(data = mydf2, aes(x, predicted), lty=2) +
  geom_ribbon(data = mydf2, aes(ymin = conf.low, ymax = conf.high),
```

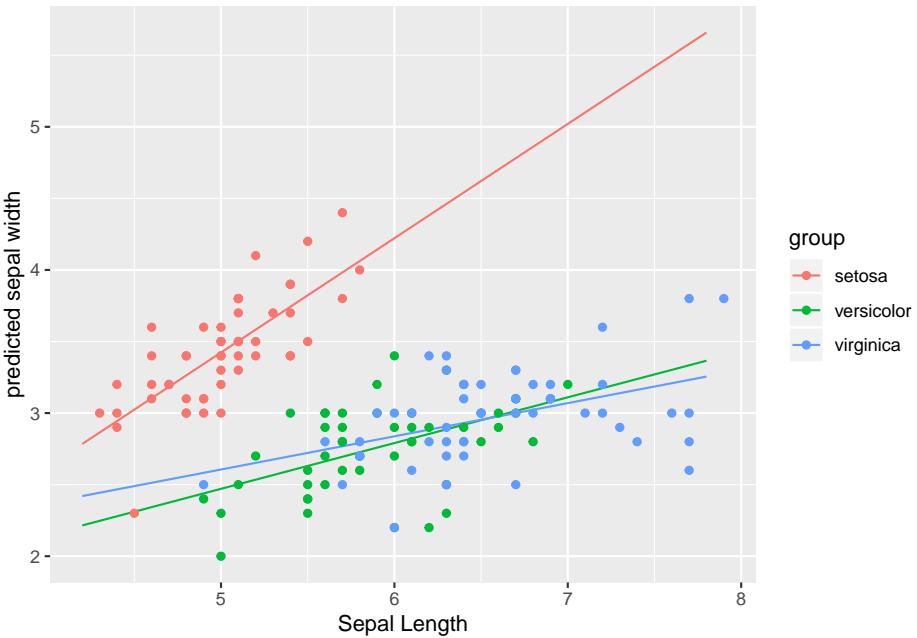
68 CHAPTER 6. VÄHIMRUUTUDE MEETODIGA FITITUD MUDELITE TÖÖVOOG – LM()

```
alpha = 0.5, fill="lightgrey") +
geom_point(data=iris, aes(Sepal.Length, Sepal.Width, color=Species)) +
xlab("Sepal Length") +
ylab("predicted sepal width")+
theme_classic()
```



terms argument võtab kuni 3 muutujat, neist 2 peavad olema fak-tormuutujad ja 3 muutuja korral tekib tabelisse veerg nimega facet, mille abil saab tulemused facet_wrap()-ga välja plottida.

```
mydf <- ggpredict(m3, terms = c("Sepal.Length", "Species"))
ggplot(mydf, aes(x, predicted)) +
  geom_line(aes(color=group)) +
  geom_point(data=iris, aes(Sepal.Length, Sepal.Width, color=Species)) +
  xlab("Sepal Length") +
  ylab("predicted sepal width")
```



Nii saab sisestada üksikuid parameetritväärtsusi ja neile ennustusi teha:

```
(mydf1 <- ggpredict(m3, terms = c("Sepal.Length [5, 22]", "Species [setosa, versicolor]")))
#>
#> # Predicted values of Sepal.Width
#> # x = Sepal.Length
#>
#> # Species = setosa
#> x predicted std.error conf.low conf.high
#> 5      3.42     0.039    3.35     3.5
#> 22     17.00    1.876   13.32    20.7
#>
#> # Species = versicolor
#> x predicted std.error conf.low conf.high
#> 5      2.47     0.08     2.31     2.63
#> 22     7.91    1.21     5.53    10.28
```

4. Võrdleme mudeleid

1. Eeldus - kõik võrreldavad mudelid on fititud täpselt samade andmete peal.
2. Eeldus (ei ole vajalik AIC meetodi puhul) - tegemist on nn nested mudelitega. Nested mudel tähendab, et kõik väiksema mudeli liikmed on olemas ka suuremas mudelis.

70CHAPTER 6. VÄHIMRUUTUDE MEETODIGA FITITUD MUDELITE TÖÖVOOG – LM()

Mudelite võrdlus ANOVA-ga (ainult nested mudelid)

```
tidy(anova(lm1, lm2, m3))
#> Warning: Unknown or uninitialized column: 'term'.
#> # A tibble: 3 x 6
#>   res.df   rss     df sumsq statistic  p.value
#>   <dbl> <dbl> <dbl> <dbl>    <dbl>      <dbl>
#> 1    148  27.9     NA NA        NA     NA
#> 2    147  15.4     1 12.5     169.  4.83e-26
#> 3    144  10.7     3  4.71     21.2  2.06e-11
```

Mudelite võrdlus AIC-ga

```
AIC(lm1, lm2, m3)
#>   df   AIC
#> lm1 3 179.5
#> lm2 4  92.1
#> m3   7  43.3
```

AIC (Akaike Informatsiooni Kriteerium) on number, mis püüab tabada mõistlikku tasakaalu mudeli fiti valimiandmetega ja parsinoomia vahel. Väiksema AIC-ga mudel on eelistatud suurema AIC-ga mudeli ees (samas, AIC-l kui ühel arvul puudub tõlgendus).

Probleem AIC-i taga on selles, et parem fit valimiandmetega võib tähendada mudeli ülefittimist (ja seega halvemat mudelit). Kuna ülefittimise tõenäosus kasvab koos mudeli keerukusega (parameetrite arvuga), eelistame võimalikult lihtsat mudelit, mis samas seletaks võimalikult suure osa valimiandmete varieeruvusest.

Chapter 7

Andmete transformeerimine

Lineaarsed transformatsioonid võivad hõlbustada mudeli koefitsientide tõlgendamist (näit. skaala millimeetritest meetritessse, tsentreerimine, standardiseerimine). Mittelineaarsed transformatsioonid (logaritmimine, jms) muudavad mudeli fitti ja võivad olla kasulikud mudeli aditiivsuse/lineaarsuse parandamisel. Oluline on mõista, et transformeeritud andmetega mudeliteid tuleb tõlgendada transformeeritud skaalas. Seega, kui algasel skaalal pole muud tõlgendust, kui et väärtsused on monotoonilised (näiteks suurem number on alati tähtsam kui väiksem number), siis sobib meile sama hästi iga lineaarne transformatsioon sellest skaalast (näiteks ruutjuure võtmine vms). Bioloogias enamasti asjad nii lihtsad ei ole ja seetõttu keskendumme siin paremini tõlgendatavatele transformatsioonidele.

7.1 Logaritmimine

Kui muutujal saavad olla ainult positiivsed väärtsused, siis on logaritmimine vahest hea mõte. Enne logaritmima asumist paeb andmetest kaotama ka nullid, näiteks asendades need mingi väikese positiivse arvuga. Logaritmilises skaalas andmetele fititud mudelite β koefitsiendid peaaegu alati < 1 .

Miks ja millal muutujaid logaritmida

1. Muutuja(te) logaritmimine muudab muutujate vahelised suhted mittele lineaarseks, samas säilitades mudeli lineaarsuse. Ja vastupidi, kui tavalisest meetrilises skaalas juhtuvad additiivse mudeli muutujate vahelised seosed olema mittele lineaarsed, siis x-muutuja(te) logaritmimine võib need muuta lineaarseks, ja sellega päästa ühe olulisema lineaarse regressiooni eelduse (vt ka ptk 4.2).

2. Logaritmimine on hea, kui soovite y ja x muutuja omavahelist sõltuvust tõlgendada üle suhtelise muutuse ehk muutuse protsendi. Kui algses skaalas on β koefitsiendi loomulik tõlgendus additiivne: x -i kasv 1 ühiku võrra ennustab y -i kasvu β ühiku võrra, siis naturaallogaritmitud x -i korral on üks loomulikest multiplikatiivsetest tõlgendustest: x -i kasv 1 ühiku võrra ennustab y -i muutust ... protsendi võrra. Additiivsel juhul me liidame ja lahutame, multiplikatiivsel juhul aga korrutame ja jagame.
3. Muutuja logaritmimine võib viia selle muutuja lähemale normaaljaotusele (lognormaaljaotuse logaritm on normaaljaotusega). Algsest paremale kaldu jaotuse äärmuspunktid võivad regressioonile liiga suurt kaalu omada, milline probleem sageli kaob logaritmimisel.
4. Kui mudeli residuaalid on ümber nulli tugevalt paremale poole kiivas jaotusega, siis andmete logaritmimine võib need normaliseerida. Samuti siis, kui residuaalide sd on proporsionaalne fititud väärtsusega (st CV, mitte SD, on konstantne) ja siis, kui te ususte, et residuaalid peegeldavad multiplikatiivseid vigu.
5. Kui y ja x -i vaheline sõltuvus on eksponentsiyalne.
6. Y -muutuja logaritmimine võib aidata hetoroskedastilisuse vastu.
7. Teaduslik teoria võib indikeerida logaritmimist. Näit pH on log skaalas.
8. Logaritmimine võib lihtsustada mudelite, vähendades interaktsiooniliikmete arvu.

Mudeli fiti kvaliteedi koha pealt pole vahet, millist logaritmi te kasutate – erinevused on “pelgalt” mudeli koefitsientide tõlgendustes. Naturaallogaritmitud $\log(x)$ andmete peal fititud mudeli korral on algses lineaarses skaalas tõlgendatav logaritmitud andmete peal fititud β , aga log-skaalas muutujate väärtsused ei tähenda peale vaadates suurt midagi. Vastupidiselt on kümnendlogaritmitud $\log_{10}(x)$ või kahendlogaritmitud $\log_2(x)$ andmed log skaalas tõlgendatavad, aga mitte neil fititud β lineaarses skaalas. Igal juhul eelistavad loodusteadlased kasutada \log_2 ja \log_{10} skaalasi, mida on mugavam otse log-skaalas tõlgendada. \log_2 skaalas vastab üheühikuline muutus kahekordsele muutusele algses skaalas ja anti-logaritm on $2^{\log_2(x)}$. \log_{10} skaalas vastab üheühikuline muutus 10-kordsele muutusele algses skaalas ja anti-logaritm on $10^{\log_{10}(x)}$.

Naturaallogaritmitud andmetega töötamine

Järgnevalt õpetame naturaallogaritmitud andmetega fititud mudelite β koefitsiendite tõlgendamist algses, meetrilises skaalas ja suhtelises protsendiskaalas.

Naturaallogaritmi alus on $e \approx 2.71828$ ja sellel on järgmised matemaatilised omadused:

1. $\log(e) = 1$

2. $\log(1) = 0$
3. $\log(A^r) = r * \log(A)$
4. $e^{\log(A)} = A$ ehk $\exp(\log(A)) = A$ ehk $2.72^{\log(A)} \approx A$
5. $\log(AB) = \log A + \log B$
6. $\log(A/B) = \log A - \log B$
7. $\exp(AB) = \exp(A)^B$
8. $\exp(A + B) = \exp(A)\exp(B)$
9. $\exp(A - B) = \exp(A)/\exp(B)$

Lineaarsel regressioonil saab log-transformatsiooni kasutada kolmel erineval viisil:

- $y = \alpha + \beta x$ — lineaarne mudel (transformeerimata)
- $y = \alpha + \beta * \log(x)$ — lineaar-log mudel (transformeeritud on prediktor(id))
- $\log(y) = \alpha + \beta x$ — log-lineaar mudel (transformeeritud on y-muutuja)
- $\log(y) = \alpha + \beta * \log(x)$ — log-log mudel (transformeeritud on y ja x muutujad)

Lineaarses mudelis $y = \alpha + \beta x$, annab β selle, mitu ühikut muutub Y keskväärtus, kui X muutub ühe ühiku võrra.

Lineaar-log mudelis jäab kehtima sama β tõlgendus, mis ülalpool, ainult et log-ühikutes. Seega viib logx-i muutus ühe log ühiku võrra y keskväärtuse muutusele β ühiku võrra (see kehtib muidugi ka log2 ja log10 skaalades).

Kui me juba kasutasime naturaallogaritmimist, siis tahame ilmselt tõlgendust pigem muutuse protsendina ja/või algsetes meetrilistes skaalas (\log_2 ja \log_{10} ei võimalda selliseid mugavaid tõlgendusi):

- β on oodatud y muutus, kui x kasvab *ex* korda.
- Kui β on väike, siis saab seda tõlgendada kui suhtelist erinevust. Näiteks, kui $\beta = 0.06$, siis 1 ühikuline x-i muutus viib u 6%-sele y muutusele. Sedamõõda kuidas β kaugeneb nullist (näiteks $\beta = 0.4$), hakkab selline hinnang tõsiselt alahindama tegelikku x-i mõju y väärtsusele.
- Oodatud y muutus kui x kasvab p protsendi on $\beta * \log([100 + p]/100)$. Näit, kui x kasvab 10% võrra (ehk kui korrutame x-i 1.1-ga), siis $\log(110/100) = 0.095$ ja 0.095β on oodatud y muutus.

Log-lineaarse mudeli korral,

- kui x kasvab 1 ühiku võrra, siis oodatud y väärtsus kasvab $\exp(\beta)$ korda.
- Kui x kasvab c ühiku võrra, siis oodatud y väärtsus kasvab $\exp(c\beta)$ korda.

- Kui β on väike, siis $100 * \beta$ vastab y protsentuaalsele muutusele juhul kui x muutub 1 ühiku võrra (kui $\beta = 0.06$, siis x-i muutus 1 ühiku võrra viib y-i 6% tõusule).

Log-log mudeli korral on tõlgendus oodatud y-i muutus protsentides kui x muutub mingi protsendi võrra. Sellist suhet kutsutakse ökonomeetrias elastiliseks ja $\log x$ -i β koefitsient on "elastilisus."

- Kui me korrutame x-i e-ga, siis korrutame oodatud y-i väärtsuse $\exp(\beta)$ -ga.
- Et saada y suhtelist muutust, kui x kasvab p protsendi, arvuta $a = \log([100 + p]/100)$ ja siis võta $\exp(a\beta)$.

7.2 Standardiseerimine

Kui prediktor x_1 on mõõdetud näiteks eurodes ja prediktor x_2 aastates, siis on meil fititud koefitsientidele b_1 ja b_2 peale vaadates raske öelda, kumb mõjutab y-muutuja väärust rohkem. Kuna euro ühik on palju granuleeritum kui aasta, siis võib ka väga väike nullist erinev b_1 omada mudeli seisukohast suuremat tähtsust kui suhteliselt suur b_2 .

$$x.z = (x - \text{mean}(x)) / \text{sd}(x)$$

Sellisel viisil standardiseeritud andmete keskväärtus on 0 ja $sd = 1$. Seega on kõik predikorid samas skaalas ja me mõõdame efekte sd ühikutes. See lubab võrrelda algselt erinevas skaalas prediktoreid. Intercept tähendab nüüd keskmist ennustust, juhul kui kõik prediktorid on fikseeritud oma keskväärtustel.

Kui mudel sisaldab lisaks pidevatele prediktoritele ka binaarseid prediktoreid, siis on kasulikum standardiseerida üle $2 \times SD$, jätkes binaarsed muutujad muutmata.

$$x.z2 = (x - \text{mean}(x)) / (2 * \text{sd}(x))$$

Nüüd tähendab 1 ühikuline muutus efekti -1 SD-st kuni 1 SD-ni üle keskväärtuse.

Korrelatsioon üle regressiooni ja regressioon keskmisele

Kui standardiseerime nii y kui x-i

$$x <- (x - \text{mean}(x)) / \text{sd}(x) y <- (y - \text{mean}(y)) / \text{sd}(y)$$

siis $y \sim x$ regressiooni intercept = 0 ja tõus on sama, mis x ja y vaheline korrelatsioonikoefitsient r. Seega jäab tõus alati -1 ja 1 vahelle.

Siit tuleb ka seletus nähtusele, mida kutsutakse regressiooniks keskmisele (*regression to the mean*). Fakti, et y on sellises mudelis alati 0-le lähemal kui x, kutsutaksegi regressiooniks keskmisele. Näiteks, kui olete 20 cm keskmisest pikem ja pikkuse päritavus on 0.5, siis on oodata, et teie järglased on keskeltläbi

10 cm võrra keskmisest pikemad (ja teist lühemad). Selle pseudo-põhjusliku nähtuse avastas Francis Galton.

7.3 Tsentreerimine

$x.c1 = x - \text{mean}(x)$ annab keskväärtuseks nulli aga jätab varieeruvused algsesse skaalasse. Näiteks interaktsioonimodelite koefitsiendid on otse tõlgendatavad tsentreeritud prediktorite korral.

Teine võimalus on tsentreerida mõnele teaduslikult mõistlikule väärtusele. Näiteks IQ-d saab tsentreerida 100-le ($x - 100$).

7.4 Mudeli koefitsientide transformeerimine

Ilma interaktsioonideta mudeli korral saab sama tulemuse, mis prediktoreid tsentreerides, kui me reskaleerime tavalises skaalas fititud mudeli koefitsiendid, korrutades iga β oma prediktori kahekordse sd-ga ($\beta_x = \beta \times 2 \times sd(x)$). Nende β_x -de pealt näeb iga muutuja suhtelist tähtsust mudelis.

Teine võimalus beta koefitsientide transformeerimiseks, mis ei eelda prediktorite normaalsust, on korrutada betad läbi vastavate muutujate interkvartiilsete range-dega (IQR).

Hoitatus: standardiseeritud koefitsiente ei tohi kasutada, et võrrelda samade prediktorite mõju erinevate andmete peal fititud sama struktuuriga mudelile.

7.5 Pidev või diskreetne muutuja?

Tavaliselt on mõistlik fittida mudel pidevale y muutujale ka siis, kui tahame lõpuks tõlgenada tulemusi diskreetsel skaalal. Pidev muutuja sisaldab rohkem informatsiooni ja seetõttu on meil lootust saada parem fit. Erandiks on sellised pidevad x muutujad, mille mõju y -le on mittelineaarne (näiteks vanuse mõju suremusele). Siin on vahest mõistlik konverteerida pidev muutuja faktormuutujaks ja saada hinnang näiteks igale vanuseklassile eraldi.

Additiivne mudel eeldab, et emaste ja isaste tõusud on võrdsed, mistõttu need fititakse sama beta koefitsiendiga, mis tähendab, et x_2 mõju pidelvale prediktorile x_1 on, et me saame mudeli ennustusena faktori n tasemele vastavad n paralleelset sirget, milles igaüks näitab pideva x_1 seost pideva y -ga erinevatel x_2 tasemetel. 0/1 kodeeringus regressorite interceptid annavad nende paralleelsete sirgete vahelised kaugused üldise-intercepti poolt antud faktori taseme ennustuses.

7.6 mitmese regressiooni üldised printsiibid

1. võta sisse kõik teaduslikku huvi pakkuvad muutujad ja viska välja muutujad, mille kohta sul pole põhust arvata, et nad võiksid y väärtsi mõjutada.
2. kontrolli, ega muutujate vahel ei esine väga tugevaid korrelatsioone (kollineaarsus). Kui jah, siis kombineeri kollineaarsed muutujad üheks või transformeerि neid või viska mõni muutuja välja.
3. muutujad, mis ei varieeru, ei oma ka regressioonis mõju.
4. tugeva mõjuga muutujate puhul võib olla vajalik sisse tuua nende muutujate interkatsioond (vt ptk ...).
5. muutujad, mida sa reaalselt mõõtsid, ei pruugi olla need muutujad, mis mudelisse lähevad – näiteks arvuta kehamassiindeks mõõdetud muutujate põhjal.
6. kui pidevad prediktorid transformeerida log skaalasse, siis on lineaarsesse mudelisse pandud efektid multiplikatiivsed, mitte aditiivsed.

Chapter 8

Veamudel

```
library(tidyverse)
library(brms)
library(broom)
```

8.1 Lihtne varieeruvuse mudel

Oletame, et me oleme mõõtnud nelja patsienti ja saanud tulemuseks 1.2, 2.12, 1.4 ja 8.34. Kuidas me oma valimit iseloomustame ja kas me peaksime 4. tulemuse, kui kahtlase, välja viskama? Arvatavasti tahaksime saada hinnangut kõige tõenäolisemale mõõtetulemusele patsientide populatsioonis ehk siis keskmise või tüüpilise patsiendi väärtsusele. Ja lisaks ka hinnangut patsientide vahelise varieeruvuse määrale. Meid võib ka huvitada võrrelda patsientide ja tervete inimeste varieeruvust. Esmapilgul tundub see lihtsa ülesandena, mis ei vaja mudeldamist – lihtsalt arvutame aritmeetilise keskmise ja standardhälbe ja meil on mõlemad hinnangud olemas. Aga tegelikult oleme probleemi ees, millele pole ühte õiget lahendust.

Kui me viskame 4. tulemuse välja, siis tuleb meie keskmise kuhugi 1.5 kanti, muidu aga piirkonda, mille lähedal meil ei ole ühtegi andmepunkti. Samuti annaks sd arvutus üsna erinevad tulemused. Kumb võimalus siis valida? Selleks peame ikkagi otsustama, kuidas modelleerida oma andmed. Sõltuvalt looduslikust protsessist, mis need andmed genereeris, võiks andmete mudel olla näiteks normaaljaotus, lognormaaljaotus vms. Kui valime normaaljaotuse, millise õlad langevad väga kiiresti, siis on vaid väike tõenäosus kohata tervelt veerandit oma andmepunktidest nõnda kaugel teistest, mis annab argumendi selle punkti eemaldamiseks. Aga lognormaaljaotuse korral, mille õlg laskub palju aeglasmalt, on tõenäosus 4. mõõtmisest isegi kaugemal olevaid andmeid kohata palju su-

urem ja seega peaksime selle andmepunkti sisse jätma. Erinevat tüüpi mudelitel on erinevad parameetrid, millele saab andmete põhjal väärtsusi otsida. See, et normaaljaotuse parameetrit μ saab meie näites arvutada aritmeetilise keskmise kaudu, ei tähenda, et ka teiste mudelite korral peaksime sama lokatsiooni-parameetrit fittima (või et neil mudelitel üldse oleks lokatsiooniparameeter). Sarnased lood on muidugi ka varieeruvust iseloomustava parameetriga.

Statistiklist mudelit saab kasutada mitmel moel.

1. Mudel toob sisse lisainformatsiooni andmete jaotuse kuju kohta, mida valimiajandmetes endis ei pruugi sisalduda, ja mis tõstab meie järelustete kvaliteeti (või langetab seda, kui valisime kehva mudeli).
2. Võrreldes erinevat tüüpi mudelite sobivust andmetega ning omades aimu protsesside kohta, mida üks või teine mudel võiks adekvaatselt kirjeldada, on võimalik teha järeldusi loodusliku mehanismi kohta, mis genereeris andmed, mille põhjal mudelid fititi.
3. Me võime fititud mudeli põhjal teha ennustusi, ehk genereerida uusi andmeid *in silico*.

Niisiis lihtne mudel andmetele: μ ehk aritmeetiline keskmine kui hinnang kõige tõenäosemale väärtsusele. See on deterministlik nn *protsessimudel*, kus samad valimiväärtused annavad alati sama ja ühese tulemuse. Statiline mudel sisaldab endas nii protsessimudelit kui tõenäosuslikku nn *varieeruvuse mudelit* (ajaloolistel põhjustel kutsutakse seda sageli veamudeliks), mis tuleb sisse tõenäosusjaotuse kujul

$$dnorm(\mu, \sigma)$$

Selle mudeli on võimalik ümber sõnastada (seda seeläbi üldistades) lihtsa regressioonivõrrandina $y = b_0$, kusjuures $\mu = b_0$ ehk andmete keskväärtus võrdub regressioonisirge interceptiga. Asendades saame

$$y \sim dnorm(b_0, \sigma)$$

Tilde \sim tähistab seose tõenäosuslikkust, ehk seda, et y muutuja ennustuslikud väärtsused tömmatakse juhuvalimina normaaljaotusest, mis omakorda on fititud empiiriliste väärtsuste (ehk valimi) põhjal.

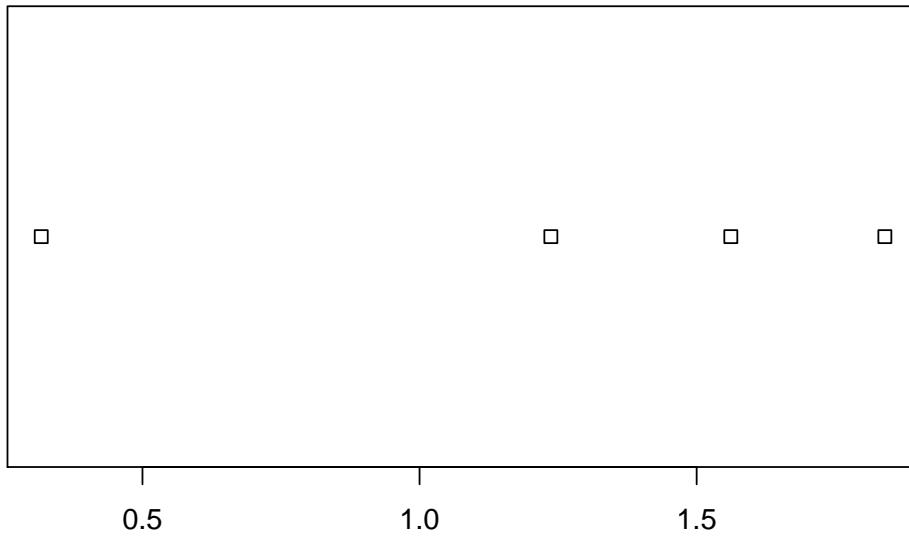
Seega on meil normaaljaotuse keskväärtus võimalik leida aritmeetilise keskmisena või samaväärsetel väljimruutude meetodiga, mis paneb keskväärtuse kohta, kus keskväärtuse ja iga andmepunkti vahelise kauguste ruutude summa tuleb minimaalne. Väljimruutude meetod on üldisem, sest töötab ka järgmises peatükis, kus me asendame μ terve regressioonivõrrandiga kujul $y = b_0 + b_1x_1 + b_2x_2 + \dots + b_ix_i$ (protsessimudel). Ja kui meie regressioonivõrrandid lähevad mittelineaarseks ja väljimruutude meetod nende fittimisel enam ei tööta, siis veel üldisem meetod, Bayesi teoreem, töötab ikka.

Kuigi aritmeetiline keskmine ja vähimruutude meetod annavad sama hinnangu lokatsiooniparameetrile, ei ütle need midagi sigma kohta. Samas Bayesi meetod annab hinnangu (koos usaldusintervalliga) mõlemale parameetrile.

Normaaljaotus mudeldab lokalisatsiooniparameetrit mu populatsiooni tüüpilise või keskmise liikme hinnanguna ja varieeruvusparameetrit sigma populatsiooni liikmete vaheliste erinevuste määra hinnanguna.

Arvutame lihtsa mudeli läbi vähimruutude meetodiga ja Bayesi meetodiga

```
set.seed(1234321)
andmed <- tibble(a= rnorm(4))
plot(andmed)
```



```
mean(andmed$a); sd(andmed$a)
#> [1] 1.24
#> [1] 0.662
```

Vähimruutude meetodit rakendab lm() funktsioon

```
lm(a~1, data = andmed) %>% broom::tidy()
#> # A tibble: 1 x 5
#>   term      estimate std.error statistic p.value
#>   <chr>      <dbl>     <dbl>      <dbl>    <dbl>
#> 1 (Intercept)  1.24      0.331     3.74   0.0333
```

Ja Bayesi brms::brm()

```
(Bayes_mudel <- brm(a~1, data = andmed) %>% broom::tidy())
#> # A tibble: 2 x 5
#>   term      estimate std.error lower upper
```

```
#>   <chr>      <dbl>    <dbl> <dbl> <dbl>
#> 1 b_Intercept 1.24     0.684 0.196 2.25
#> 2 sigma        1.27     1.24  0.461 3.02
```

Nagu näete, lm() fitib ainult mu parameetri, samas kui me Bayesi meetodit kasutades saame hinnangu (koos usalduspiiridega) kahele parameetrile: mu ehk intercept ja sigma ehk sd.

Meie poolt simuleeritud andmed tulevad normaaljaotusega populatsioonist, mille $\mu = 0$ ja $sd = 1$. Kumbki meetod ei luba meile null-intercepti sest andmeid on vähe ja need on juhusliku valimivea tõttu kallutatud. See-eest sigma hinnang, mille Bayes meile annab on küll laiavöitu (ikka sellepäast, et meil on vähe andmeid), aga vähemalt hõlmab endas õiget väärust.

8.2 protsessimudel ja varieeruvuse mudel lineaarses regressioonis

Kui mudel $kaal = b_0 + b_1 \text{ pikkus}$ ennustab, et 160 cm inimene kaalub keskmiselt 80 kg, siis protsessi mudel ei ütle, kui suurt pikkusest sõltumatut kaalude varieeruvust võime oodata 160 cm-ste inimeste hulgas. Selle hinnangu andmiseks tuleb mudelile lisada varieeruvusekomponent, sageli normaaljaotuse kujul, mis modelleerib üksikute inimeste kaalude varieeruvust (mitte keskmise kaalu varieeruvust) igal mõeldaval pikkusel.

Bioloogid, erinevalt füüsikutest, usuavad, et valimisisene andmete varieeruvus on tingitud pigem bioloogilisest varieeruvusest kui mõõtmisveast, aga loomulikult sisaldub selles ka mõõtmisviga.

Kuidas varieeruvuskomponent lineaarsesse mudelisse sisse tuua? Ilma varieeruvuskomponendita mudel:

$$y = b_0 + bx$$

ennustab y-i keskväärtust erinevatel x-i väärustel.

Varieeruvuskomponent:

$$y \sim dnorm(\mu, \sigma)$$

kus μ (mu) on mudeli poolt ennustatud keskväärtus ja σ ($sigma$) on mudeli poolt ennustatud standardhälve ehk varieeruvus andmepunktide tasemel. Varieeruvuskomponent on keskväärtuse ehk mu ennustus endiselt deterministlik ja σ töötab originaalsel andmetasemel, mitte keskväärtuste tasemel. See võimaldab protsessimudeli varieeruvuskomponenti sisse kirjutada lihtsalt mu ümber defineerides:

$$\mu = b_0 + bx$$

mis tähendab, et

$$y \sim dnorm(b_0 + b_1x, \sigma)$$

See ongi sirge mudel koos varieeruvuskomponendiga. Seega on selle lineaarsel regressioonimudelil kolm parameetrit: intercept b_0 , tõus b_1 ja "veaparameeter" σ . Sellist mudelit on mõistlik fittida Bayesi teoreemi abil. Bayesi meetodiga fititud mudel, mida kutsutakse posteerioriks, näitab, millised kombinatsioonid nendest kolmest parameetrist usutavalt koos esinevad, ja millised mitte. Seega on fititud 3 parameetriga bayesi mudel 3-dimensionaalne töenäosusjaotus (3D posteerior). Muidugi saame ka üksshaaval välja plottida kolm 1D posteriori, millega igaüks iseloomustab üht parameetrit ning on kollapseeritud üle kahe ülejäänud parameetri. Edaspidi õpime selliste mudelitega töötama.

Kõik statistilised mudelid on töenäosusmudelid ning sisaldavad varieeruvuskomponenti.

Kuna erinevalt lokatsiooniparameetrist, ei aja me mudelis sigmat lahku vastavalt x-i väärustele, siis meie varieeruvusmudel (ja enamus veamudeleid, millega me edaspidi töötame) modelleerib igale x-i väärusele (kaalule) sama suure y-i suunalise varieeruvuse (pikkuste sd). Suurem osa statistikast kasutab eeldusi, mida keegi pärtselt töe pähe ei võta, aga millega on arvutuslikus mõttes lihtsam elada. Siiski, 19. peatükis õpime, kuidas loobuda sellest eeldusest.

Enimkasutatud veamudel on normaaljaotus

Alustuseks simuleerime lihtsate vahenditega looduslikku protsessi, mille tulemusel tekib normaaljaotus.

Oletame, et bakteri kasvukiirust mõjudavad 12 geeni, mille mõjud võivad olla väga erineva tugevusega, kuid mille mõjude suurused ei sõltu üksteisest. Seega nende 12 geeni mõjud kasvukiirusele liituvad. Järgnevas koodis võtame 12 juhuslikku arvu 1 ja 100 vahel (kasutades `runif()` funktsiooni). Need 12 arvu näitavad 12 erineva geeni individuaalse mõjude suurusi bakteritüve kasvukiirusele. Meil on seega kuni 100-kordsed erinevused erinevate geenide mõjude suuruste vahel. Seejärel liidame need 12 arvu. Nüüd võtame uue 12-se valimi ja kordame eelnevad. Me teeme seda 10 000 korda järjest ja plotime saadud 10 000 arvu (10 000 liitmistehte tulemust) tihedusfunktsionina.

```
kasv <- replicate(10000, sum(runif(12, 1, 100)))
p <- ggplot(tibble(kasv), aes(kasv)) + geom_density()
p
```

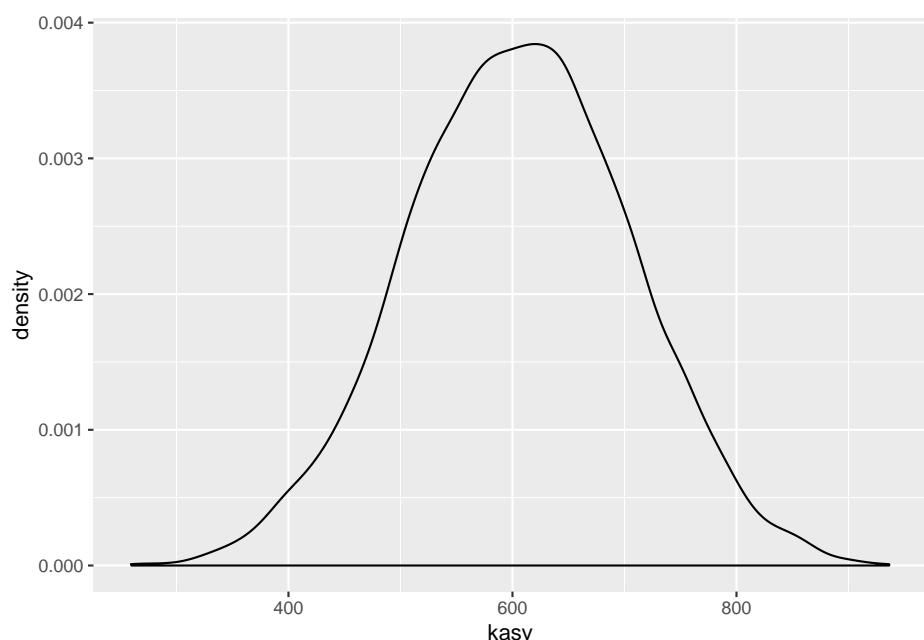


Figure 8.1: Normaaljaotus tekib sõltumatutest efektidest. Kümne tuhande $N = 12$ suuruse juhuvalimi summa tihedusdiagramm.

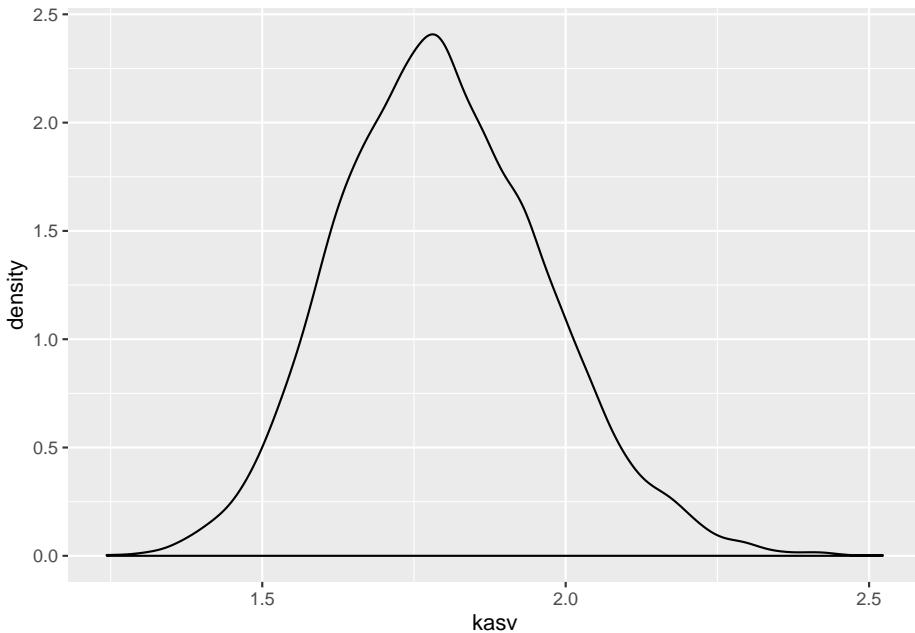


Figure 8.2: Normaaljaotus tekib väikestest sõltuvatest efektidest. Kümne tuhande $N = 12$ suuruse juhuvalimi korrutiste tihedusdiagramm. Ühegi geenि mõju ei domineeri teiste üle.

Selles näites võrdub iga andmepunkt 10 000st ühe bakteritüve kasvukiiruse mõõtmisega. Seega, antud eelduste korral on bakteritüvede kasvukiirused normaaljaotusega.

Nüüd vaatame, mis juhtub, kui 12 geenи mõjud ei ole üksteisest sõltumatud. Kui 12 geenи on omavahel vastasmõjudes, siis nende geenide mõjud korruutuvad, mitte ei liitu. (Korrutamise pole ainus viis, kuidas vastasmõjusid modeelerida, küll aga kõige levinum.) Kõigepealt vaatleme juhtu, kus 12 geenи on kõik väikeste mõjudega ning seega mitte ühegi geenи mõju ei domineeri teiste üle. Seekord genreerime 12 juhuslikku arvu 1 ja 1.1 vahel. Siin tähendab arv 1.1 kasvu tõusu 10% võrra. Seejärel korruutame need 12 arvu, misjärel kordame eelnevat 10 000 korda.

```
kasv <- replicate(10000, prod(runif(12, 1, 1.1)))
p %+%
  tibble(kasv)
```

Tulemuseks on jällegi normaaljaotus. Selles näites olid üksikud interakteeruvad geenid ükshaaval väikeste mõjudega ja ühegi geenи mõju ei domineerinud teiste üle. Mis juhtub, kui mõnel geenil on kuni 2 korda suurem mõju kui teisel?

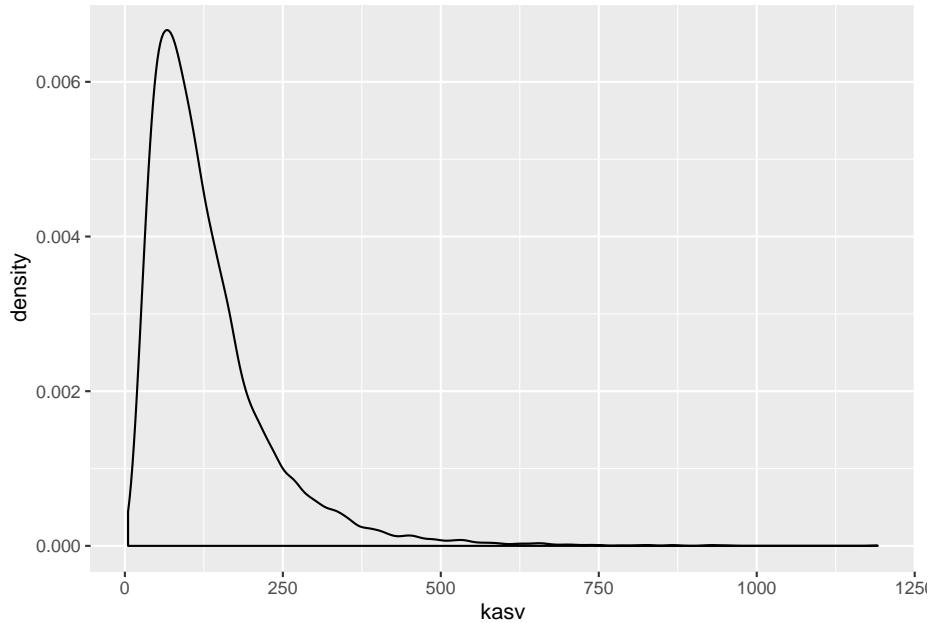


Figure 8.3: Lognormaaljaotus tekib suurematest sõltuvatest efektidest. Kümne tuhande $N = 12$ suuruse juhuvalimi korrutiste tihedusdiagramm. Mõnel geenil on kuni 2 korda suurem mõju kui teisel.

```
kasv <- replicate(10000, prod(runif(12, 1, 2)))
p %>% tibble(kasv)
```

Nüüd on tulemuseks log-normaaljaotus. Mis teie arvate, kas teie poolt uuritavat tunnust mõjutavad faktorid, mis omavahel ei interakteeru või kui interakteeruvad, on kõik ühtlaselt väikeste efektidega? Või on tegu vastasmõjudes olevate faktoritega, millest osad on palju suuremate mõjudega, kui teised? Ühel juhul eelistate te normaaljaotust, teisel juhul peate õppima töötama ka lognormaaljaotusega.

Kui me vaatame samu andmeid logaritmilises skaalas, avastame, et need andmed on normaaljaotusega. See ongi andmete logaritmimise mõte.

```
kasv <- replicate(10000, log10(prod(runif(12, 1, 2))))
p %>% tibble(kasv) + labs(x = "kasv, log10")
```

Normaaljatuse avistas Gauss (1809), aga nime andis sellele Francis Galton (1860ndatel), kuna antropoloogilised mõõtmised “normaalselt” järgisid “vigade seadust”, mille ta nimetas “Normaalseks jaotuste kurviks”.

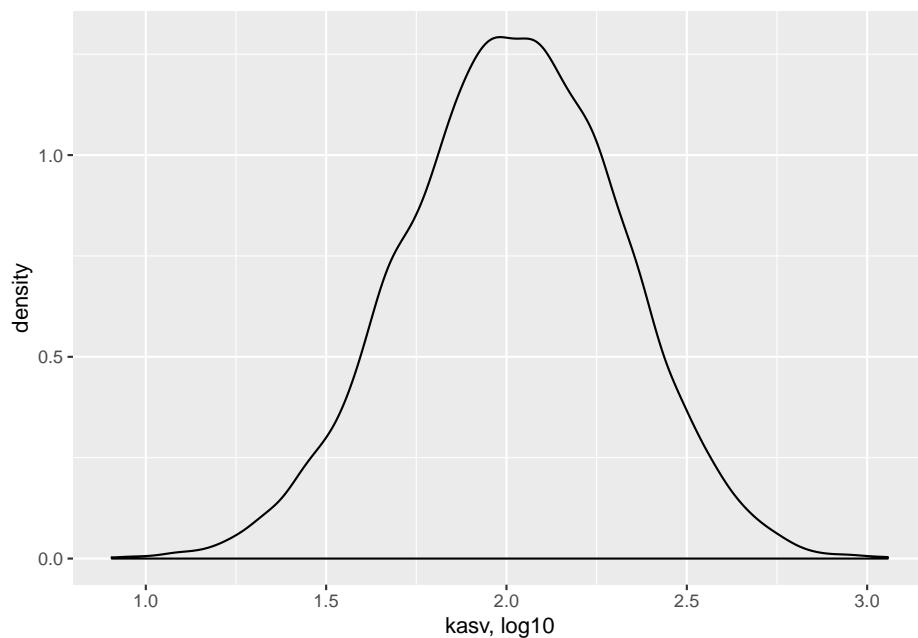


Figure 8.4: Logaritmilises skaalas lognormaalsed efektid on normaaljaotusega. Kümne tuhande N = 12 suuruse juhuvalimi korrutiste tihedusdiagramm. Mõnel geenil on kuni 2 korda suurem mõju kui teisel.

Normaaljaotuse mudel väikestel valimitel

Oletame, et meil on kolm andmepunkti ning me usume, et need andmed on juhuslikult tõmmatud normaaljaotusest või sellele lähedasest jaotusest. Normaaljaotuse mudelit kasutades deklareerime, et me usume, et kui oleksime olnud vähem laisad ja 3 mõõtmise asemel sooritanuks 3000, siis need mõõtmised sobituksid piisavalt hästi meie 3 väärтuse peal fititud normaaljaotusega. Seega, me usume, et omades 3 andmepunkti me teame juba umbkaudu, millised tulemused me oleksime saanud korjates näiteks 3 miljonit andmepunkti. Oma mudelist võime simuleerida ükskõik kui palju andmepunkte.

Aga pidage meeles, et selle mudeli fittimiseks kasutame me ainult neid andmeid, mis meil päriselt on — ja kui meil on ainult 3 andmepunkti, on tõenäoline, et fititud mudel ei kajasta hästi tegelikkust.

Kuidas panna skeptik uskuma, et statistilised meetodid töötavad halvasti väikestel valimitel? Järgnevalt illustreerime seda ühe võimaliku valimiga paljudest, mis on tõmmatud imaginaarset populatsioonist, mille parameetreid me teame. Me tõmbame 3-se valimi ning üritame selle valimi põhjal ennustada selleasama populatsiooni struktuuri. Kuna tegemist on simulatsiooniga, teame täpselt, et populatsioon, kust me tõmbame oma kolmese valimi, on normaaljaotusega, et tema keskväärtus = 0 ja et tema $sd = 1$. Seega saame võrrelda oma ennustust populatsiooni tõeliste parameetrväärtustega. Me fitime oma valimiandmetega 2 erinevat mudelit: normaaljaotuse ja Studenti t jaotuse.

Siin saame hinnata mudelite fitte jumala positsioonilt, võrreldes fititud mudelite jaotusi "tõese" sinise jaotusega. Mõlemad mudelid on süstemaatiliselt nihutatud väiksemate väärтuse poole ja alahindavad varieeruvust. t jaotuse mudel on oodatult paksemate sabadega ja ennustab 0-st kaugele palju rohkem väärтusi kui normaaljaotuse mudel. Kuna me teame, et populatsioon on normaaljaotusega, pole väga üllatav, et t jaotus modeleerib seda halvemini kui normaaljaotus.

Igal juhul, mõni teine juhuvalim annaks meile hoopis teistsugused mudelid, mis rohkem või vähem erinevad algsest populatsioonist.

Mis juhtub kui me kasutame oma normaaljaotuse mudelit uute andmete simuleerimiseks? Kui lähedased on need simuleeritud andmed populatsiooni andmetega ja kui lähedased valimi andmetega, millega me normaaljaotuse mudeli fittisime?

```
# tõmbame 3 juhuslikku arvu normaaljaotusest, mille keskväärtus = 0 ja sd = 1.
dfr <- tibble(sample_data = rnorm(3))
dfr <- summarise_at(dfr, "sample_data", c("mean", "sd"))
dfr
#> # A tibble: 1 x 2
#>   mean    sd
#>   <dbl> <dbl>
#> 1 0.0654 0.808
# simuleerime 1000 uut andmepunkti fititud mudelist
```

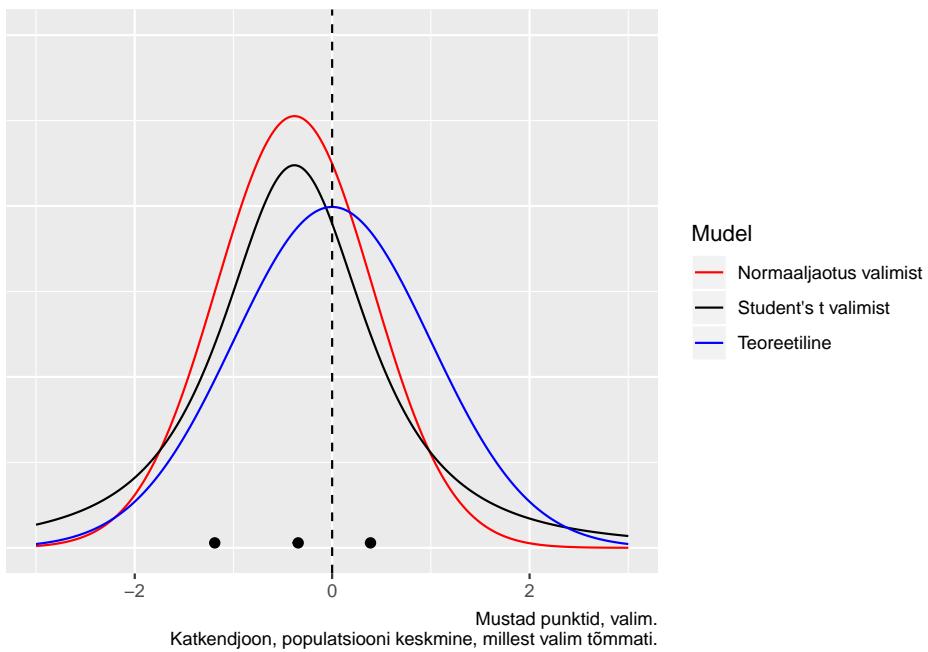


Figure 8.5: Juhuvalim normaaljaotusest, mille keskmine = 0 ja $sd = 1$ ($n=3$; andmepunktid on näidatud mustade munadena). Sinine joon - populatsioon, milles tõmmati valim; punane joon - normaaljaotuse mudel, mis on fititud valimi andmetel; must joon - Studenti t jaotuse mudel, mis on fititud samade andmetega. Mustad punktid, valim. Katkendjoon, populatsiooni keskmise, milles tõmmati.

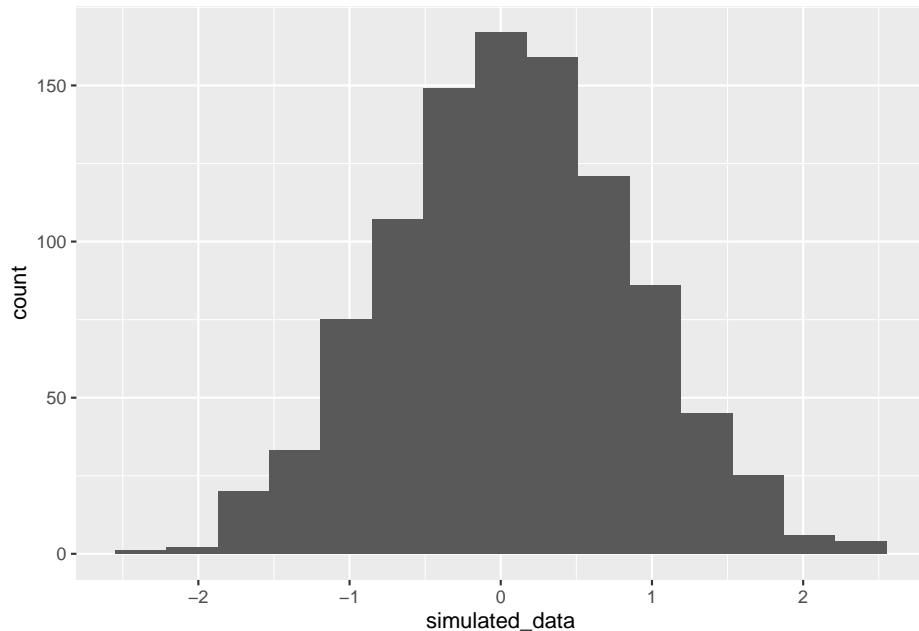


Figure 8.6: Kasutame fititud mudeleid uute andmete simuleerimiseks.

```
simulated_data <- rnorm(1000, dfr$mean, dfr$sd)
# arvutame simuleeritud andmete keskmise ja sd ning joonistame neist histogrammi
ggplot(tibble(simulated_data), aes(simulated_data)) +
  geom_histogram(bins = 15)
```

Nagu näha, igati ootuspäraselt on uute (simuleeritud) andmete keskväärtus ja SD väga sarnased algsete andmete omale, mida kasutasime mudeli fittimisel. Kahjuks ei ole need aga kaugeltki nii sarnased algsele jaotusele, mille kuju me püüame oma andmete ja mudeli pealt ennustada. Seega on meie mudel üle-fittitud, mis tähendab, et ta kajastab liigsett neid valimi aspekte, mis ei peegelda algse populatsiooni omadusi. Loomulikult ei vasta ükski mudel päriselt tegelikkusele. Küsimus on pigem selles, kas mõni meie mudelitest on piisavalt hea, et olla kasulik. Vastus sellele sõltub, milleks plaanime oma mudelit kasutada.

```
mean(simulated_data > 0)
#> [1] 0.535
mean(simulated_data > 1)
#> [1] 0.116
```

Kui populatsiooniväärtustest on 50% suuremad kui 0, siis mudeli järgi vaevalt 32%. Kui populatsiooniväärtustest on 16% suuremad kui 1, siis mudeli järgi vaevalt 4%. See illustreerib hästi mudeli kvaliteeti.

```
sim_t <- rstudent_t(1000, 2, dfr$mean, dfr$sd)
mean(sim_t > 0)
#> [1] 0.516
mean(sim_t > 1)
#> [1] 0.189
```

Samad ennustused t jaotusest on isegi paremad! Aga kumb on ikkagi parem mudel populatsioonile?

Normaaljaotuse ja lognormaaljaotuse erilisus

Normaaljaotus ja lognormaaljaotus on erilised sest

- (1) kesksest piirteoreemist (*central limit theorem*) tuleneb, et olgu teie valim ükskõik millise jaotusega, paljudest valimitest arvutatud **aritmeetilised keskmised** on alati enam-vähem normaaljaotusega. See kehtib enamuse andmejaotuste korral, kui $n > 30$. Selle matemaatilise töe peegeldus füüsikalisse maailma on “elementaarsete vigade hüpotees”, mille kohaselt paljude väikeste üksteisest sõltumatute juhuslike efektide (vigade) summa annab tulemuseks normaaljaotuse.

Paraku enamus bioloogilisi mõõtmisi annavad tulemuseks eranditult mitte-negatiivseid väärtsusi. Sageli on selliste väärtsuste jaotused ebasiümmeetrilised (v.a. siis, kui $cv = sd/mean$ on väike), ja kui nii, siis on meil sageli tegu lognormaaljaotusega, mis tekkib log-normaalsete muutujate korrutamisest. Siit tuleb Keskne piirteoreem 2, mille kohaselt suvalise jaotusega muutujate **geomeetrilised keskmised** on enam-vähem lognormaaljaotusega, ning elementaarsete vigade hüpotees 2: Kui juhuslik varieeruvus tekib paljude juhuslike efektide korrutamisel, on tulemuseks lognormaaljaotus. Lognormaaljaotusega väärtsuste logaritmimine annab normaaljaotuse.

- (2) Nii normaal- kui lognormaaljaotus on maksimaalse entroopiaga jaotused. Entroopiat vaadeldakse siin informatsiooni & müra kaudu — maksimaalsete entroopiaga süsteem sisaldab maksimaalselt müra ja minimaalselt informatsiooni (vastavalt Shannoni informatsiooniteooriale). See tähendab, et väljaspool oma parameetrite tuunitud väärtsusi on normaal- ja lognormaaljaotused minimaalselt informatiivsed. Normaaljaotusel ja lognormaaljaotusel on kummagil kaks parameetrit, μ ja σ (ehk keskmise ja standardhälve), mille väärtsused fikseerime üheselt jaotuse ehk mudeli kuju, lisades sinna minimaalselt muud (sooviamtut) informatsiooni. Teised maksimaalsete entroopiaga jaotused on näiteks eksponentsiyalne jaotus, binoomjaotus, bernoulli jaotus, poissoni jaotus.

Kui meil on tegu nullist suuremate andmetega, on andmete logaritmimine sageli hea mõte. Logaritmitud andmete pealt arvutatud

keskmise ja sd eksponentimine annab meile geomeetrilise keskmise ($\exp(\mu) = \mu_{\text{geom}}$) ja multiplikatiivse sd ($\exp(\sigma^2) = \sigma^2_{\text{mult}}$).

Kui me fitime lognormaaljaotust andmetega, siis fititud koefitsiidid mu ja sd tuleb eksponentida, et saada geomeetriline keskmise ja multiplikatiivne sd.

$\mu_{\text{geom}} \times \sigma^2_{\text{mult}} \dots \mu_{\text{geom}}/\sigma^2_{\text{mult}}$ annab vahemiku, kuhu jäab 68% lognormaalsetest andmetest ja $\mu_{\text{geom}} \times 2\sigma^2_{\text{mult}} \dots \mu_{\text{geom}}/2\sigma^2_{\text{mult}}$ annab vahemiku, kuhu jäab 96% andmetest (just nagu additiivne sd tõõtab aritmeetilise additiivse keskmisega normaalsete andmete korral).

Maksimaalsel entroopial põhineb normaaljaotuse ja lognormaaljaotuse sage kasutamine Bayesi statistikas prioritenaga, sest me suudame paremini kontrollida, millist informatsiooni me neisse surume. Esimesel kesksel piirteoreemil seevastu põhineb kogu sageduslik statistika (vt ptk 8.).

8.2.1 Normaaljaotuse ja lognormaaljaotuse võrdlus

Normaaljaotus

1. Normaaljaotusega ehk normaalsete juhuslike muutujate liitmine annab normaalse summa. Lineaarsed kombinatsioonid $Y = \alpha + \beta_1 X_1 + \beta_2 X_2$ jäavad normaalseks.
2. Normaalsete muutujate aritmeetilised keskmised on normaaljaotusega.
3. Keskne piirteoreem: mitte-normaalsete muutujate aritmeetilised keskmised on enam-vähem normaaljaotusega.
4. Elementaarsete vigade hüpotees: kui juhuslik varieeruvus on paljude juhuslike mõjude summa, on tulemuseks normaaljaotus.
5. Additiivne regressioonimodel (normaalne töepära) viib additiivsetele viigadele (residuaalidele), mis omakorda viib konstantsele varieeruvusele (SD-le). Vead on normaaljaotusega.

lognormaaljaotus

1. lognormaalsete juhuslike muutujate korrutamine annab lognormaalse korrutise.
2. Longnormaalsete muutujate geomeetrilised keskmised on lognormaaljaotusega.
3. Keskne piirteoreem: mitte-lognormaalsete muutujate geomeetrilised keskmised on enam-vähem lognormaaljaotusega
4. Elementaarsete vigade hüpotees: kui juhuslik varieeruvus on paljude juhuslike mõjude korrutis, on tulemuseks lognormaaljaotus

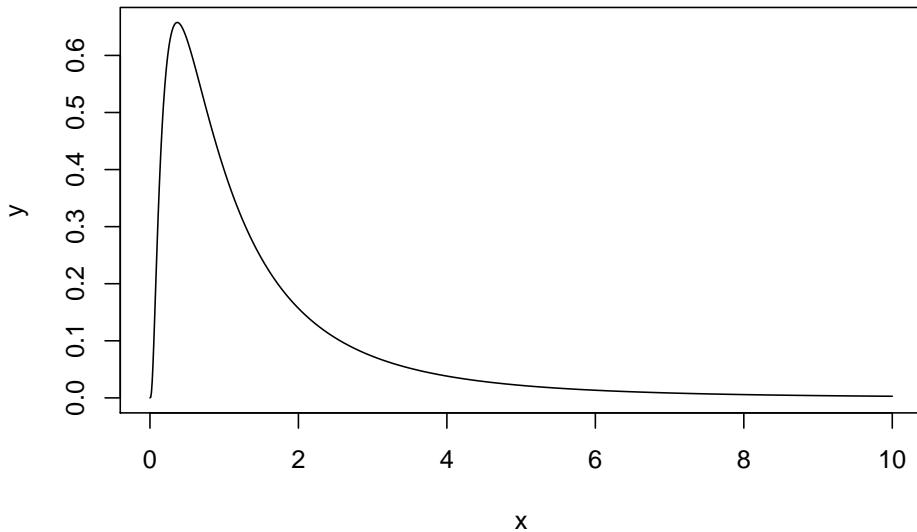
5. multiplikatiivne regressioonimudel (lognormaalne töepära) viib multiplikatiivsete vigadeni ja konstantsele suhtelisele varieeruvusele (CV-le). Vigade jaotus on ebasümmeetiline.

Seega võime lognormaaljaotust kutsuda ka multiplikatiivseks normaaljaotuseks.

8.3 Teised veamudelid

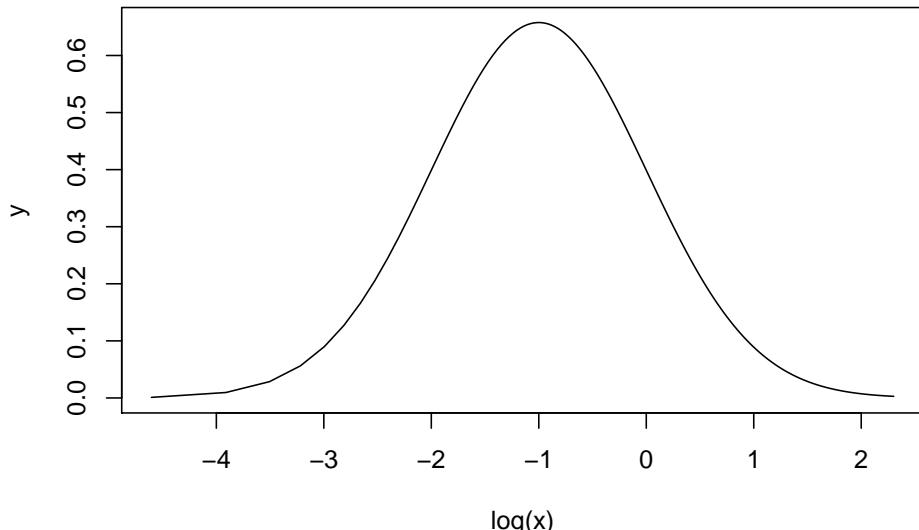
8.3.1 Lognormaaljaotus

```
x <- seq(0, 10, length.out = 1000)
y <- dlnorm(x)
plot(x, y, typ = "l")
```



Seda jaotust, mis ei ulatu kunagi teisele poole nulli, iseloomustab, et x-i logaritmimine annab tulemuseks normaaljaotuse.

```
plot(log(x), y, type = "l")
```



Lognormaaljaotuse keskväärtus, standardhälve, mood ja mediaan:

$$\text{keskväärtus} = \exp(\mu + 1/2 \times \sigma^2)$$

$$sd = \exp(\mu + 1/2 \times \sigma^2) \times \sqrt{\exp(\sigma^2) - 1}$$

$$\text{mood} = e^{\mu - \sigma^2}$$

$$\text{mediaan} = e^\mu$$

Siin on siis μ ja σ arvutatud logaritmitud andmete pealt.

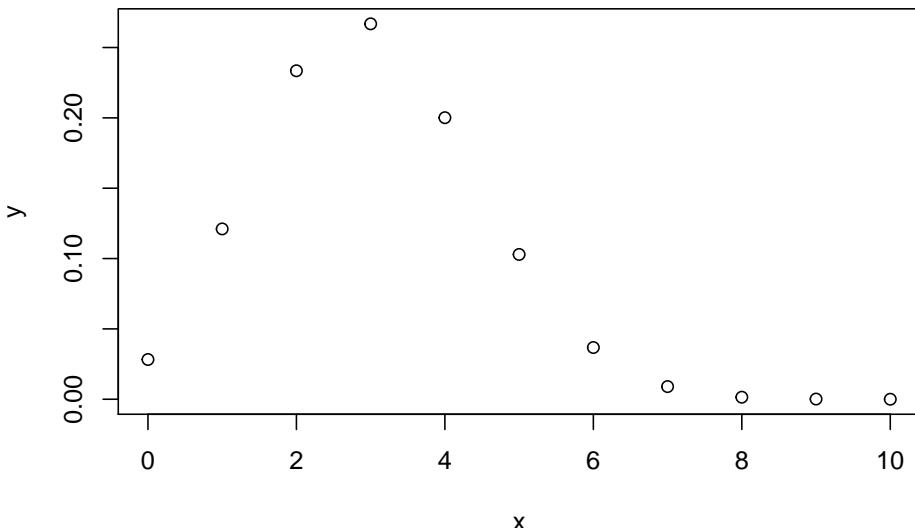
8.3.2 Binoomjaotus

Kui teil on binaarne muutuja (sellel saab olla ainult kaks väärust, näiteks sees/väljas, 1/0), mis kajastab sõltumatuid sündmusi, siis modelleerib seda binoomjaotus $y \sim \text{Binomial}(n, p)$. Kus n on edukate sündmuste arv ja p on nende suhteline sagedus ($p = n / N$, kus N on kõikide sündmuste kopguarv). Sõltumatud sündmused on sellised, kus ühe sündmuse esinemise järgi ei saa ennustada teise sündmuse esinemist (st puudub korrelatsioon sündmuste esinemise vahel). Tehniliselt on binoomjaotusel veel omadus, et valim võetakse replacementiga, mis tähendab, et iga sündmus pannakse populatsiooni tagasi, kus seda saab uuesti valimisse tömmata. Siit tuleb, et binoomjaotuse mudel kehtib päris maailmas mõõndustega ja et seda mudelit on kindlam kasutada siis, kui $N >> n$. Kui N on suur, siis meenutab binoomjaotus normaaljaotust (läheneb selle kujule).

```

n <- 10 # sündmuste koguarv
x <- seq(0, n) # kõik võimalikud õnnestumiste arvud 10st sündmusest
p <- 0.3 # 30% õnnestumisi (sagedus)
y <- dbinom(x, n, p)
plot(x, y)

```



$$\text{keskväärtus} = N \times p$$

Kui Np võrdub täisarvuga, siis mediaan = mood = keskväärtus

$$sd = \sqrt{N \times p(1 - p)}$$

Standardviga proportsioonile $p = \sqrt{\frac{p(1-p)}{N}}$ See standardviga (*standard error*) on teiste sõnadega standardhälve meie hinangule proportsiooni väärtsusele. Kui $n = 0$ või $N - n = 0$, siis on selline SE arvutus eksitav.

8.3.3 Poissoni jaotus

See jaotus modelleerib üksikuid haruldasi ja sõltumatuid diskreetseid sündmusi, mille arvu me saame üles lugeda. Näiteks surmi ajaühiku kohta või pommitabamusi pindalaühiku kohta. Poissoni jaotus on binoomjaotuse erijuht. Lisaeeldused on, et sündmuste toimumise sagedus ei muutu, et kaks sündmust ei saa toimuda täpselt samal ajal/kohas, et sündmuse toimumise töenäosus on proportsionaalne intervalli pikkusega/suurusega (ajas või ruumis) ja et $N \gg n$.

Kui keskmise sündmuste arv intervallis on λ (lambda), siis

$$P(k \text{ events in interval}) = e^\lambda \times \frac{\lambda^k}{k!}$$

Oodatud väärthus = variance = λ

$$sd = \sqrt{\lambda}$$

Millal kasutada Poissoni jaotust, ja millal binoomjaotust? Kui iga andmepunkti saab vaadelda kui edukate katsete arvu suhet kõikide katsete arvule, siis kasuta binoomjaotust/logistilist regressiooni. Kui aga andmepunkti väärthusel pole loomulikku piiri (see on lihtsalt mingit tüüpi siundumste arv), kasuta Poissoni/logaritmilist regressiooni.

Chapter 9

EDA — eksploratoorne andmeanalüüs

```
library(tidyverse)
library(corrgram)
library(psych)
library(skimr)
```

Kui ühenumbiline andmete summeerimine täidab eelkõige kokkuvõtliku kommunikatsiooni eesmärki, siis EDA on suunatud teadlasele endale. EDA eesmärk on andmeid eelkõige graafiliselt vaadata, et saada aimu 1) andmete kvaliteedist ja 2) lasta andmetel kõneleda “sellisena nagu nad on” ja sugereerida uudseid teaduslike hüpoteese. Neid hüpoteese peaks siis testima formaalse statistilise analüüsiga abil (ptk järel dav statistika). Näiteid erinevate graafiliste lahendustega kohta vt graafika peatükist.

EDA: mida rohkem graafikuid, seda rohkem võimalusi uute mõtetega tekkeks!

EDA on rohkem kunst kui teadus selles mõttes, et teil on suur vabadus küsida selle abil erinevaid küsimusi oma andmete kohta. Ja seda nii tehnilisest aspektist lähtuvalt (milline on minu andmete kvaliteet?), kui teaduslike küsimusi küsides (kas muutuja A võiks põhjustada muutusi muutujas B?).

Mõned üldised soovitused võib siiski anda.

1. alusta analüüsi tasemest, kus andmed on kõige inforikkamad — toorangete plottimisest punktidena. Kui andmehulk ei ole väga massiivne, näitab see hästi nii andmete kvaliteeti, kui ka võimalikke sõltuvussuhteid erinevate muutujate vahel.

Millised korrelatsioonid võiksid andmetes esineda?

Correlogram of Iris dataset

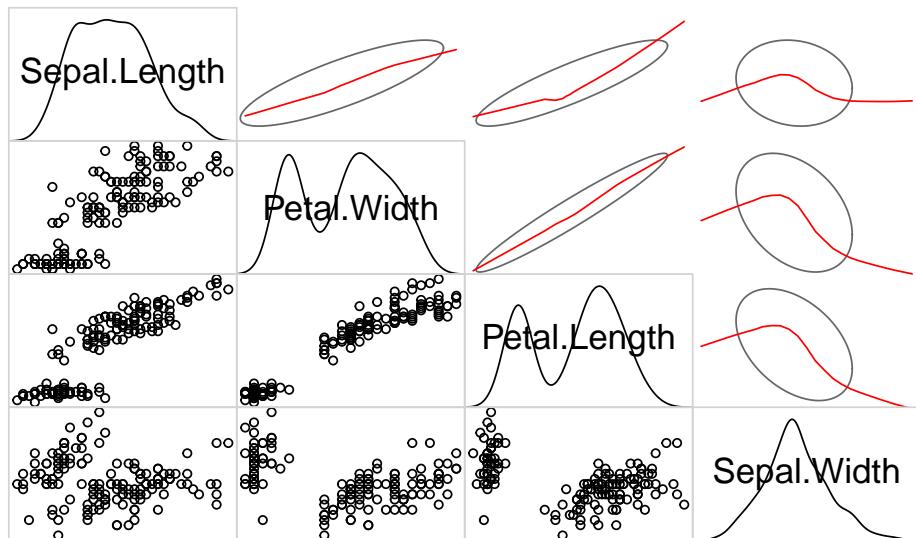


Figure 9.1: Korrelatstioonimaatriks joonisena.

```
corrgram(iris,
          order = TRUE,
          lower.panel = panel.pts,
          upper.panel = panel.ellipse,
          diag.panel = panel.density,
          main = "Correlogram of Iris dataset")
```

2. vaata andmeid numbrilise kokkuvõttena.

```
psych::describe(iris)
#>           vars   n  mean    sd median trimmed  mad min max range skew
#> Sepal.Length    1 150 5.84 0.83    5.80    5.81 1.04 4.3 7.9   3.6  0.31
#> Sepal.Width     2 150 3.06 0.44    3.00    3.04 0.44 2.0 4.4   2.4  0.31
#> Petal.Length    3 150 3.76 1.77    4.35    3.76 1.85 1.0 6.9   5.9 -0.27
#> Petal.Width     4 150 1.20 0.76    1.30    1.18 1.04 0.1 2.5   2.4 -0.10
#> Species*        5 150 2.00 0.82    2.00    2.00 1.48 1.0 3.0   2.0  0.00
#>                  kurtosis   se
#> Sepal.Length    -0.61 0.07
#> Sepal.Width      0.14 0.04
#> Petal.Length    -1.42 0.14
#> Petal.Width     -1.36 0.06
```

```
#> Species*      -1.52 0.07

skimr::skim(iris)
#> Skim summary statistics
#> n obs: 150
#> n variables: 5
#>
#> -- Variable type:factor -----
#>   variable missing complete   n n_unique          top_counts
#>   Species        0     150 150           3 set: 50, ver: 50, vir: 50, NA: 0
#>   ordered        FALSE
#>
#> -- Variable type:numeric -----
#>   variable missing complete   n mean    sd  p0  p25  p50  p75  p100
#>   Petal.Length    0     150 150 3.76 1.77 1   1.6 4.35 5.1  6.9
#>   Petal.Width     0     150 150 1.2  0.76 0.1 0.3 1.3  1.8  2.5
#>   Sepal.Length    0     150 150 5.84 0.83 4.3 5.1 5.8  6.4  7.9
#>   Sepal.Width     0     150 150 3.06 0.44 2   2.8 3   3.3  4.4
```

Siin pööra kindlasti tähelepanu tulpadele min ja max, mis annavad kiire võimalusi outliereid ära tunda. Kontrolli, kas andmete keskmised (mediaan, mean ja trimmed mean) on üksteisele piisavalt lähedal — kui ei ole, siis on andmete jaotus pika õлага, ja kindlasti mitte normaalne. Kontrolli, kas erinevate muutujate keskväärtused ja hälbed on teaduslikus mõttes usutavas vahemikus. Ära unusta, et ka väga väike standardhälve võib tähendada, et teie valim ei peegelda bioloogilist varieeruvust populatsioonis, mis teile teaduslikku huvi pakub. NB! selles psych::describe() funktsiooni väljundis on mad läbi korrutatud konstantida 1.4826, mis toob selle vääruse lähemale sd-le. Seega on mad siin sd robustne analoog — kui mad on palju väiksem sd-st, siis on karta, et muutujas on outliereid.

3. kontrolli NA-de esinemist oma andmetes VIM paketi abil või käitsi (vt esimene ptk). Kontrolli, et NA-d ei oleks tähistatud mingil muul viisil (näiteks 0-i või mõne muu numbriga). Kui vaja, rekodeeri NAd. Mõtle selle peale, millised protsessid loodusnesid võiksid genreerida puuduvaid andmeid. Kui NA-d ei jaotu andmetes juhuslikult, võib olla hea mõte andmeid imputeerida (vt hilismaid ptk, bayesiaanlik imputeerimine). Näiteks, kui ravimiuringust kukuvad eeskätt välja patsiendid, kellel ravim ei tööta, on ilmselt halb mõte nende patsientide andmed lihtsalt uuringust välja vistata (muidugi, kui te ei esinda kasumit taotleva ettevõtte huve). Kui NA-d jaotuvad juhuslikult, mõtle sellele, kas sa tahad NA-dega read tabelist välja visata, või hoopis osad muutujad, mis sisaldavad liiga palju NA-sid, või mitte midagi välja vistata. NB! NA-dega andmed ei sobi hästi regresioniks.
4. Kui andmeid on nii palju, et üksikute andmepunktide vaatlemine paneb

pea valutama, siis järgmine informatiivsuse tase on histogramm.

5. kui tahame kõrvuti vaadata paljude erinevate muutujate varieeruvust ja keskväärtusi, siis on head valikud joyplot, violin plot, ja vähem hea valik (sest ta kaotab andmetest rohkem infot) on boxplot. Kui meil on vaid 2-4 jaotust, mida võrrelda, siis saab mängida histogramme facetisse või üksteise otsa pannes (vt ptk graphics).
6. Tulpdiagramm on hea valik siis, kui tahate kõrvuti näidata proportsioonide erinevust. Näiteks, kui meil on 3 liiki kalu, millest igas on erinevas proportsioonis parasiidid, võime joonistada 3 tulpa, millest igas on näidatud ühe kalaliigi parasiitide omavaheline proportsioon.
7. Tulpdiagramm on hädaga pooleks kasutuskõlblik, kui iga muutuja kohta on vaid üks number, mida plottida. Kuigi, siin on meil parem võimalus — Cleveland plot. Olukorras, kus te tahate plottida valimi keskväärtust ja usalduspiire või varieeruvusnäitajat (sd, mad), on olemas selgelt paremad meetodid kui tulpdiagramm. Samas, ehki tulpdiagrammide kasutamine teaduskirjanduses on pikas langustrendis, kasutatakse neid ikkagi liiga palju just sellel viisil.
8. Ära piirdu muutuja tasemel varieeruvuse plottimisega. Teaduslikult on sageli huvitavam mimte muutuja koosvarieerumine. Järgmistes peatükkides modelleerime seda formaalselt regresionanalüüsits aga alati tasub alustada lihtsatest plottidest. Scatterplot on lihtne viis kovarieeruvuse vaatamiseks.
9. Kui erinevad muutujad on mõõdetud erinevates skaalades (ühikutes), siis võib nende koosvarieeruvust olla kergem võrrelda, kui nad eelnevalt normaliseerida (kõigi muutujate keskväärtus = 0, aga varieeruvus jäab algsesse skaalasse) või standardiseerida (kõik keskväärtused = 0-ga ja sd-d = 1-ga). Normaliseerimine: arvuta igale valimi väärusele: `mean(x) - x`; standardiseerimine: `(mean(x) - x) / sd(x)`.
10. Visualiseeringu valik sõltub valimi suurusest. Väikse valimi korral ($N < 10$) boxploti, histogrammi vms kasutamine on lihtsalt rumal. Ära mängi lolli ja ploti parem punkti kaupa.
 - $N < 20$ - plotti iga andmepunkt eraldi (`stripchart()`, `plot()`) ja keskmine või mediaan.
 - $20 > N > 100$: `geom_dotplot()` histogrammi vaates
 - $N > 100$: `geom_histogram()`, `geom_density()` — nende abil saab ka 2 kuni 6 jaotust võrrelda
 - Mitme jaotuse kõrvuti vaatamiseks, kui $N > 15$: `geom_boxplot()`, or `geom_violin()`, `geom_joy()`
11. Nii saab plottida multiplikatiivse sd:

```
#> Warning: Ignoring unknown parameters: bins
```

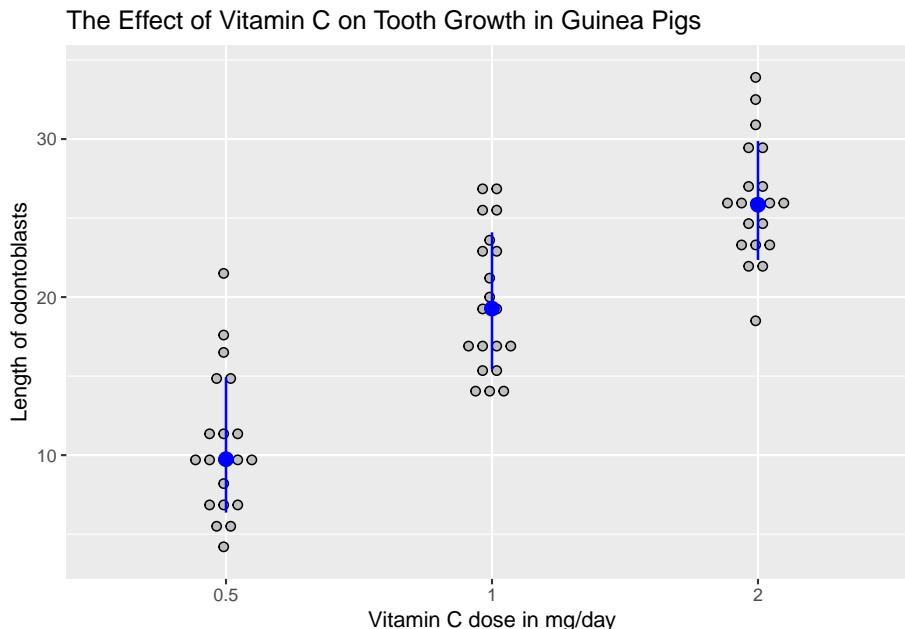


Figure 9.2: Multiplikatiivse sd joonistamine.

9.1 EDA kokkuvõte

1. Andmepunktide plottimine säilitab maksimaalselt andmetes olevat infot (nii kasulikku infot kui müra). Aitab leida outliereid (valesti sisestatud andmeid, valesti mõõdetud proove jms). Kui valim on väiksem kui 20, piisab täiesti üksikute andmepunktide plotist koos mediaaniga. Dot-plot ruulib.
2. Histogramm – kõigepealt mõõtskaala ja seejärel andmed jagatakse võrdse laiusega binnidesse ja plotitakse binnide kõrgused. Bin, kuhu läks 20 andmepunkti on 2X kõrgem kui bin, kuhu läks 10 andmepunkti. Samas, bini laius/ulatus mõõteskaalal pole teile ette antud – ja sellest võib sõltuda histogrammi kuju. Seega on soovitav proovida erinevaid bini laiusi ja võrrelda saadud histogramme. Histogramm sisaldab vähem infot kui dot plot, aga võimaldab paremini tabada seaduspärasid & andmejaotust & outliereid suurte andmekoguste korral.
3. Density plot. Silutud versioon histogrammist, mis kaotab infot aga toob vahest välja signaali müra arvel. Density plotte on hea körvuti vaadelda joy ploti abil.
4. Box-plot — sisaldab vähem infot kui histogramm, kuid neid on lihtsam

kõrvuti võrrelda. Levinum variant (kuid kahjuks mitte ainus) on Tukey box-plot – mediaan (joon), 50% IQR (box) ja 1,5x IQR (vuntsid), pluss outlierid eraldi punktidena.

5. Violin plot — informatiivsuselt box-ploti ja histogrammi vahepeal – sobib paljude jaotuste kõrvuti võrdlemiseks.
6. Line plot — kasuta ainult siis kui nii X kui Y teljele on kantud pidev väärthus (pikkus, kaal, kontsentratsioon, aeg jms). Ära kasuta, kui teljele kantud punktide vahel ei ole looduslik mõtet omavaid pidevaid väärthusi (näiteks X teljel on katse ja kontroll või erinevad valgumutatsioonid, mille aktiivsust on mõõdetud).
7. Tulpdiagramm – Suhete võrdlemine (bar).
8. Cleveland plot on hea countide võrdlemiseks. Kui Cleveland plot mingil põhjusel ei sobi, kasuta tupldiagrammi.
9. Pie chart on proportsioonide vaatamiseks enam-vähem kõlblik ainult siis, kui teil pole vaja võrrelda proportsioone erinevates objektides. Kõik graafikud, kus lugeja peab võrdlema pindalasid, on inimmõistusele petlikud — lugeja alahindab süstemaatiliselt erinevuste suurus! Selle pärast on proportsioonide võrdlemiseks palju parem tulpdiagramm, kus võrreldavad tulbad on ühekõrgused.

Informatsiooni hulk kahanevalt: iga andmepunkt plotitud → histogramm → density plot & violin plot → box plot → tulpdiagramm standardhälvetega → cleveland plot (ilma veapiirideta)

Chapter 10

Lausearvutuslik loogika

Enne, kui siirdume järeldava statistika ja tõenäosusteooria juurde, teeme lühikese sissejuhatuse klassikalisse loogikasse, sest tõenäosusteoria ei ole lõppude-lõpuks midagi muud, kui loogika laiendus juhule, kus me ei ole kindlad selles, mida räägime. Niisiis, loogika ülesanne on modelleerida inimkeelseid lauseid (või nende sisu ehk propositioone). Keele modelleerimise läbi modelleerime me ühtlasi mõlemist, kaasa arvatud teaduslik mõlemine. Nagu ikka, ei eelda me ka siin, et mudel vastaks täpselt reaalsusele.

Igal juhul koosneb meie keelemudeli baas-süntaks sõnadest nagu “ja”, “või”, “mitte”, “kui … siis”, mida kutsume *konnektiivideks*.

Konnektiivid koos neid tähistava sümboliga:

- *not* – (\neg) – negatsioon e *negation*,
- *and* – (\wedge) – konjunksioon e *conjunction*,
- *or* – (\vee) – disjunksioon e *disjunction*,
- *if … then* – (\rightarrow) – implikatsioon e konditsionaal e *implication* e *conditional* (if – *antecedent*, then – *consequent*).

Suured tähed A, B, C, … tähistavad *atomaarseid lauseid*. Iga atomaarne lause tähistab ühte või mitut inimkeelset lauset. Loogiku jaoks pole atomaarsete lausete sisemine struktuur oluline, sest sellest ei sõltu mudelkeele lausete valiidsus.

Mudelkeele 1. tase koosneb atomaarsetest lausetest, mis on ühendatud konnektiividega (negatsioon, konjunksioon jne). Siin on juba tegemist *liitlausetega*. 2. taseme liitlaused koosnevad konnektiividega ühendatud 1. taseme lausetest, ja nii edasi lõpmatusse. Näiteks $((A \rightarrow B) \vee (B \wedge A)) \rightarrow C$ on 3-tasemeline lause, kus sulud näitavad, milliseid komponentlauseid mingi konnektiiv parajagu ühendab.

Lisaks konnektiividele sisaldab meie keelemudel tõeväärtusi: T ja F. Siinkohal eeldame, et mitmetasemeliste lausete tõeväärtused sõltuvad nende aluseks olevate atomaarsete lausete tõeväärtustest, ja mitte millegist muust. Seda eeldust kutsutakse tõetabeli printsibiks.

10.1 Tõetabel

Kui me tähistame suvalist liitlauset X-ga, siis tõetabel näeb välja nii:

```
#> # A tibble: 4 x 3
#>   A     B     X
#>   <chr> <chr> <chr>
#> 1 T     F     ""
#> 2 F     T     ""
#> 3 T     T     ""
#> 4 F     F     ""
```

Tõetabel annab kõik võimalikud kombinatsioonid atomaarsete lausete tõeväärtustest ja ütleb iga sellise kombinatsiooni kohta, kas X on tõene või väär. Seega on tõetabel loogiline diagramm X-le. Tabeli iga rida annab ühe kombinatsiooni atomaarsete lausete tõeväärtustest ja X veeru vastavale reale peaks kirjutama X-i tõeväärtuse, mis sõltub nende atomaarsete lausete tõeväärtusest sellel real ja sellest, milliste konnektiividega need on liitlausesse X ühendatud. Tabeli iga rida annab X-le unikaalse tõeväärtuse.

Kõigepealt anname tõetabeli negatsioonile, mis on unaarne konnektiiv, ehk töötab ühe lause piires

```
#> # A tibble: 2 x 2
#>   A     `not A`
#>   <chr> <chr>
#> 1 T     F
#> 2 F     T
```

Ehk sõnadega: $\neg A$ on tõene siis kui A on väär, ja vastupidi. Negatsioon ei tee muud, kui põõrab lause tõeväärtuse vastupidiseks.

10.2 Konjunktsioon

Nüüd tõetabel konjunktsioonile, mis on binaarne konnektiiv, ühendades kahte lauset.

```
#> # A tibble: 4 x 3
#>   A     B     `A and B`
#>   <chr> <chr> <chr>
```

```
#> 1 T      T      T
#> 2 T      F      F
#> 3 F      T      F
#> 4 F      F      F
```

Ehk sõnadega: $A \wedge B$ on tõene siis ja ainult siis kui A ja B on mõlemad tõesed.

Konjunktsiooni võib kasutada näiteks nii:

P1 (Premiss 1): Seda ja teist

J1 (Järeldus 1): Seda

J2: teist

Või:

P1: ei ole külm ega tuuline

J1: ei ole külm

J2: ei ole tuuline

Aga lausest $\neg(A \wedge B)$ ei saa midagi järeltada:

P1: Ma ei ole praegu Pariisis ega Tallinnas

J1: —

10.3 Disjunktsioon

Nüüd disjunktsioon. Loogikute jaoks on siin tegemist nn inklusiivse või-ga, mis tähendab, et see kehtib ka siis, kui A ja B mõlemad kehtivad.

```
#> # A tibble: 4 x 3
#>   A     B     `A or B`
#>   <chr> <chr> <chr>
#> 1 T     T     T
#> 2 T     F     T
#> 3 F     T     T
#> 4 F     F     F
```

$A \vee B$ on väär siis ja ainult siis, kui A ja B on mõlemad väärad.

Kuidas aga oleks lood ekslusiiivse disjunktsiooniga (*xor*), kus $A = T$ ja $B = T$ viivad väärale disjunktsioonile? Me ei vaja xor jaoks tingimata eraldi konnektiivi (sümbolit), sest selle töetabel langeb kokku lause

$$(A \vee B) \wedge \neg(A \wedge B)$$

tõetabeliga.

Selle arvutamiseks evaluateerime kõigepealt sisemise disjunktiooni $A \vee B$ ja konjunksiooni $A \wedge B$, seejärel negatsiooni $\neg(A \wedge B)$ ja lõpuks kogu lause (see on tabelis kaskel olev “and”).

```
#> # A tibble: 4 x 6
#>   A     B     `A or B (I)` `and (III)` `not (II)` `A and B (I)`
#>   <chr> <chr> <chr>      <chr>      <chr>      <chr>
#> 1 T     T     F           F           T
#> 2 T     F     T           T           F
#> 3 F     T     T           T           F
#> 4 F     F     F           F           F
```

Tabeli evaluateerimise järjekord on antud rooma numbritega tabeli veergude päistes.

Ja võrdluseks A xor B tõetabel

```
#> # A tibble: 4 x 3
#>   A     B     `A xor B`
#>   <chr> <chr> <chr>
#> 1 T     T     F
#> 2 T     F     T
#> 3 F     T     T
#> 4 F     F     F
```

Kuna nende tabelite läbiarvutamisel saadud tõeväärtused on identsed, on laused $(A \vee B) \wedge \neg(A \wedge B) \Leftrightarrow A \text{ xor } B$ loogiliselt ekvivalentsed. Ekvivalentsed võivad olla ka laused, mis ei koosne samadest atomaarlausetest, senikaua kui nende tõetabeli kõik read on sama tõeväärtusega (näiteks A ja $A \wedge (B \vee \neg B)$).

Disjunksiooni ja konjunksiooni on võimalik avaldada teineteise kaudu:

$$\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$$

$$\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$$

Disjunksiooni saab kasutada näiteks nii:

P1: Ei seda ega teist ($\neg(A \vee B)$)

J1: Ei seda ($\neg A$)

J2: ei teist ($\neg B$)

Või

P1: ei mitte-A ega mitte-B ($\neg(\neg A \vee \neg B)$)

J1: A

J2: B

Samas, lausest $A \vee B$ ei saa midagi järelidata:

P1: Ma olen kas Pariisis või Tallinnas

J1: —

10.4 Konditsionaal

Ja lõpuks konditsionaali $A \rightarrow B$ tõetabel

```
#> # A tibble: 4 x 3
#>   A     B     `if A then B`
#>   <chr> <chr> <chr>
#> 1 T     T     T
#> 2 T     F     F
#> 3 F     T     T
#> 4 F     F     T
```

Konditsionaal on väär siis ja ainult siis kui A on tõene ja B on väär. Vahest kipuvad inimesed nägema konditsionaali põhjusliku seose mudelina. See ei ole aga hea mõte, sest loogilised tehted eeldavad ainult koos või eraldi esinemist, mitte põhjuslikke ega ajalisi suhteid.

Veel üks oluline samasus:

$$A \rightarrow B \Leftrightarrow \neg B \rightarrow \neg A$$

Lisaks võime konditsionaali avaldada ka läbi disjunktsiooni või konjunktsiooni:

$$A \rightarrow B \Leftrightarrow \neg A \vee B \Leftrightarrow \neg(A \wedge B)$$

. Ainus põhjus, miks meil on eraldi konnektiiv nimega konditsionaal, on selle järeldusliku vormi sage kasutamine. Seega on konditsionaal loogikas sisuliselt vähetähtis mugavussümbol, mitte põhjusliku seose sügavmõtteline mudel.

Konditsionaali $A \rightarrow B$ osaline vaste tõenäosusteoorias on tingimuslik tõenäosus $P(B | A)$, mis ütleb “B tõenäosus tingimusel, et A on tõene” (vt allpool).

Konditsionaali saab kasutada näiteks nii

P1: $\neg(A \rightarrow B)$

J1: A

J2: $\neg B$

Jällegi, lausest $A \rightarrow B$ ei saa midagi järelidata A ega B kohta.

P1: $A \rightarrow B$

J1: —

10.5 Tautoloogia ja kontradiktsioon

$A \vee \neg A$ on tautoloogia, sest selle tõetabelis on X alati tõene

```
tibble(A = c("F", "T"), `A or notA` = c("T", "T"))
#> # A tibble: 2 x 2
#>   A     `A or notA`
#>   <chr> <chr>
#> 1 F      T
#> 2 T      T
```

Tautoloogiast tuleneb välistatud kolmanda seadus, mille kohaselt iga propositsioon on kas tõene või väär (ja mitte kunagi mõlemat korraga).

Seevastu $A \wedge \neg A$ on kontradiktsioon ehk iseendaga vastuoluline ehk loogiliselt vastuoluline, sest selle tõetabelis on X alati väär.

```
tibble(A = c("F", "T"), `A and notA` = c("F", "F"))
#> # A tibble: 2 x 2
#>   A     `A and notA`
#>   <chr> <chr>
#> 1 F      F
#> 2 T      F
```

Nagu juba eespool mainitud, kui tõetabelis leidub rida, kus kõik atomaarsed laused on tõesed ja X on väär, siis ja ainult siis on tegu kontradiktsiooniga. Antud juhul on selline tabeli 2. rida

10.6 loogiline argument ja valiidne järeldamine

Näiteks lause: maa on kerakujuline või kuu on juustust. Nüüd eeldame, et maa ei ole kerakujuline. Siit tuleb loogiliselt valiidne järeldus: kuu on juustust.

Ehk

P1: AvB

P2: mitte-A

J: B

Siin on meil tegemist loogilise **argumendiga**, mis koosneb kahest **premissist** (P1 ja P2) ja järeldusest (J). Premissid on laused, mille kohta me eeldame, et need on tõesed, ja järelduse me dedutseerime premissidest lähtuvalt sellest eeldusest.

Mis juhtub, kui me eeldame, et järeldus B on hoopiski väär, aga premissid mitte-A ja AvB on mõlemad tõesed? Sellisel juhul on meil tegu loogilise vastuolu e kontradiktsiooniga. Seega on premissidest dedutseeritud järeldus loogiliselt

tõsikindel; iga deduktiivne järeldus on juba peidus premissides ja ei sisalda endas uut informatsiooni.

Järelduse loogiline valiidsus ei taga selle kehtivust päris maailmas (kui tagaks, siis me elaksime vaid matemaatikast koosnevas maailmas, mille mõistmiseks poleks vaja teha empiirilisi uuringuid). Me võime sama hästi eeldada, et (P1) maa on kerakujuline või kuu on juustust e AvB, (P2) et kuu ei ole juustust e mitte-B, ja siit järeldub, et maa on kerakujuline:

AvB

mitte-B

J: A,

See järeldus on nii valiidne kui kehtiv. Aga samas empiiriliselt mitte kuigi huvitav.

Argument on **valiidne** siis ja ainult siis, kui olukord, kus kõik premissid oleksid tõesed ja järeldus oleks väär, on loogiliselt vastuoluline.

Argument on **kehtiv** (*sound*) siis ja ainult siis, kui see on valiidne ja kõik premissid on tõesed.

Argumendi valiidsus tähistab palgalt argumendi korrektset semantilist struktuuri (ehk loogilist vormi). Argumendi kehtivus tähendab, et argumendi järeldus on ka sisuliselt kehtiv ehk tõene. Valiidne järeldamine eeldab, et premissid ja järeldus on ehitatud atomaarsetest lausetest nii, et ei esine atomaarsete lausete tõeväärtuste kombinatsiooni, mis muudaks kõik premissid tõeseks ja järelduse vääraks. Kui siiski esineb selline kombinatsioon, siis oleme leidnud loogilise vastuolu ehk kontradiktsiooni ja meie järeldamismehhanism ei saa olla valiidne.

Selle näitlikustamiseks kontrollime argumendi

P1: $A \rightarrow B$

P2: $\neg A$

J: $\neg B$

valiidsust töetabeli abil:

```
tibble(A=c("T", "T", "F", "F"), B=c("T", "F", "T", "F"), `P1: if A then B` = c("T", "F", "T", "T"),
#> # A tibble: 4 x 5
#>   A     B     `P1: if A then B` `P2: notA` `J: notB`
#>   <chr> <chr> <chr>        <chr>      <chr>
#> 1 T     T     T             T         T
#> 2 T     F     F             F         T
#> 3 F     T     T             T         F
#> 4 F     F     T             T         T
```

Töetabelist on näha, et see argument ei ole valiidne, sest tabeli 3. reas on tõesed premissid ja väär järeldus. Nii lihtne see ongi. Pane tähele, et sellises töetabelis

on huvitavad ainult sellised read, kus ükski premiss pole väär ja järelalus on väär. Kõiki teisi ridu võib ignoreerida. Kuna tabeli ridade arv võrdub kaks astmes atomaarsete lausete arv, milline number kasvab atomaarsete lausete arvu kasvuga väga kiiresti, tasub seda meeles pidada.

10.7 Modus Ponens ja Modus Tollens

Bertrand Russelile, kellel on väga suured teened formaalse loogika arendamisel 20. sajandi alguses, kuulub väike nali teadusliku meetodi kohta, nagu seda nägid paljud 20. sajandi teadusfilosoofid (Russell, 1945):

If p , then q ; now q is true; therefore p is true.
 E.g. if pigs had wings then some winged animals
 are good to eat; therefore pigs have wings.
 This form of inference is called `scientific method`.

See inglise huumor näitlikustab induktiivset teadusliku mõtlemise mudelit, mis ekslikult kasutab deduktiivse lausearvutusliku süllogismi mitte-valiidset vormi. Tegemist on sedavõrd levinud eksitusega, et sellel on lausa oma ladinakeelne nimi, mida võib tõlkida kui “peale seda, järelikult selle pärast” (*Post hoc ergo propter hoc*). Selle süllogismi vähem naljakas rakendus oleks:

P1: Kui patsiendil on gripp, siis on tal (töenäoliselt) palavik [$A \rightarrow B$]

P2: palavik [B]

J1: gripp [A]

J2: töenäoliselt gripp [$P(A)$ on kõrge]

Paraku kumbki järelalus ei kehti.

Teine ja palju kavalam katse lausearvutusliku loogika abil teaduslikku mõtlemist mudeldada kuulub teadusfilosoof Karl Popperile. Et Popperi mudelit tutvustada, alustame valiidsest (ehkki mitte tingimata kehtivast) deduktiivsest argumendist ladinakeelse nimega *Modus Ponens*

P1: $A \rightarrow B$

P2: A

J: B

Ehk,

P1: kõik mehed on sead (kui mees, siis siga)

P2: Aristoteles on mees

J: Aristoteles on siga

Et modelleerida üldist ja alati kehtivat loodusseadust, mis oli Popperi jaoks teaduslik teoria par excellence, seondub selle argumendiga probleem, millest oli teadlik juba Aristoteles. Kui me tahame tõsikindlalt näidata, et kõik mehed on tõepoolest seat, siis peame minema induktiivset rada ja testima tõepoolest kõiki mehi, nii praegusi, eilasi, kui homseid selles osas, kui palju nad sigu meenutavad.

P1: 1. mees on siga

P2: 2. mees on siga

.....

Pn: n-s mees on siga

J: Kõik mehed on seat

See ei ole paraku teostatav.

Popper püüdis probleemi lahendada, tuues sisse valiidse dedukiivse argumendi vormis *Modus Tollens*:

P1: $A \rightarrow B$

P2: $\neg B$

J: $\neg A$

ehk:

P1: kõik mehed on seat

P2: Aristoteles ei ole siga

J: Aristoteles ei ole mees

Aga seda võib vaadata ka nii: Kui me eeldame, et Aristoteles siiski on mees, ja et Aristoteles ei ole siga, siis argumendi valiidsuse päästmiseks teeme järelduse, et P1 on väär (st kõik mehed ei ole tehs mitte seat). Sellisel viisil loogilise vastuolu lahendamine on täiesti lubatud ja soositud tegevus.

Seega oli Popperi retsept teadlastele (loe: füüsikutele)

1. postuleeri üldine teoria vormis: kõik X-d on Y.
2. Dedutseeri sellest mõni teaduslikult testitav alamteooria vormis x_i on Y.
3. Juhul kui me suudame empiiriliselt näidata, et see alamteooria on väär, oleme sellega dedukiivselt ümber lükanud ka üldise teoria kehtimise.

Seda skeemi illustreerib hästi Enrico Fermi tsitaat:

If your experiments succeed in provig the hypothesis,
you have made a measurement; if they fail to prove
the hypothesis, you have made a discovery.

Sellist suure teoria ümber lükkamist kitsama haardega alamteoria testimise läbi nimetatakse teoria falsifitseerimiseks. Siit tuleneb ka Popperi ettepanek teaduse ja mitte-teaduse eristamiseks: kõik teaduslikud teoriad peavad olema vähemalt põhimõtteliselt falsifitseeritavad (sest muidu ei saaks neid Popperi teadusliku mõtlemise mudeli abil ümber lükata), millest tuleneb omakorda, et mida lihtsam on teoriat falsifitseerida, seda “teaduslikum” see teoria on. Näiteks teoria, mille kohaselt igal kolmapäeval kell 14:00 sajab Ilmatsalu ilmajaamas 3 mm õlevihma, on suurepäraselt falsifitseeritav ja seega super-teaduslik. Igal juhul lõi Popper kõigepealt teadusliku mõtlemise formaalse mudeli ja teatas seejärel, et kuna see mudel töötab ainult teatud struktuuriga teoriate peal, siis kallid teadlased, palun ajage oma teoriad õigesse vormi või leppige sildiga “mitte-teaduslik”.

Teine häda oli see, et alamteoria ümber lükkamiseks viisil, mis kindlustab Modus tollensi kehtimise, peame olema absoluutset kindel, et me oleme selle päriselt ümber lükanud. Seega peame oleme täiesti kindlad, et meie katseparatuur teeb seda, mida me tahame, et mõõtmisviga ei vii meid ekslikele järeldustele jne. Popperil oli selle vastuväite osas öelda seda, et olgu peale, me peame kasutama eeldusi, mille kehtimises me ei saa kindlad olla, aga vähemalt põhimõtteliselt oleme me nõus iga sellise eelduse avama ja läbi vaatama, kui selleks peaks vajadus tekkima. Senikaua kui see on nii, on Popperi järgi tegu teadusega. Seega me eeldame, et Modus Tollens töötab nagu kellavärk, aga ainult mõtlemise mudeli piires. Tegelikus teaduslikus praktikas ei saa siiski millegile kindel olla!

Falsifitseerimise kui teadusliku mõtlemise mudeli põhiline ja ületamatu puudus on, et see töötab lausearvutusliku loogika raames, mis tähendab, et see jäab paratamtult hätta teoriatega, mis ennustavad millegi juhtumist tõenäosuslikult. Näiteks teoria, mille kohaselt suitsetamine põhjustab kopsuvähki, aga mitte igal suitsetajal (suitsetamise põhiline suremust tõstev mõju on läbi südame-haiguste, mitte vähi). Lausearvutuses ei ole ühtegi mehhanismi tõenäosuslike propositsoonidega töötamiseks ja Popperi, kes oli mõnede arvates oma põlvkonna nutikaim filosoof, 70 aastat kestnud pingutused selline mehhanism luua jooksid liiva.

10.8 Lausearvutusest tõenäosuste loogikasse

Mis juhtub, kui meie premiss ei ole mitte “kuu on tehtud juustust” vaid “kuu on võib-olla tehtud juustust”, ehk “meil on andmeid, et kuu on tehtud juustust”, ehk “kuu on tõenäoliselt tehtud juustust”? Sellisel juhul ei ole loogiline järeldus “A” ehk “A = TRUE”, vaid hoopis “võib-olla A” ehk “tõenäoliselt A” ehk “P(A) = [reaalarv 0 ja 1 vahel]”. Lausearvutuse reeglid eeldavad, et premissid on kas tõesed või väärads, ehk premisside (ja järelduste) tõenäosused tohivad omada vaid kahte väärust: 1 ja 0. Seega ei saa me siin lausearvutust rakendada ja vajame teistsugust loogikat, mis võimaldaks ebakindlate premisside põhjal teha ebakindlaid järeldusi. See tähendab, et me vajame tõenäosusteoorigat.

Kui lausearvutus töötab must-valges töene-vääär maailmas, siis tõenäosusteooria opereerib halli varjunditega. Tõenäosusteoorialt kui loogika laienduselt ootame, et see annaks meile järeldused kujul “A tõenäosus” ($P(A)$) või “A tõenäosus, juhul kui kehtib B” ($P(A \text{ I } B)$). Lisaks ootame, et alati, kui tõenäosused on fikseeritud ühe ja nulliga, annaks tõenäosusteooria välja samad järeldused kui lausearvutus. Üldiselt tahame me mõlema loogika puhul sama: konverteerida premissid parimateks võimalikeks järeldusteks, mida saaksime (küll mingil määral ja mööndustega) formaalse mudeli maailmast ka päris maailma üle kanda.

Lausearvutus on deduktiiivne süsteem, kus järelduse töesus sisaldub juba premissides. Kui loogik on töese järelduseni jõudnud, siis see järeldus on igavene – seda ei saa muuta uusi premissi või andmeid lisades. Seda omadust nimetatakse loogika monotoonilisuseks. Teisisõnu, lausearvutuslik loogika on mõtlemise mudel, mis ei sisalda kahtlusi ega isegi võimalust kahtlusteks. Selline mudel ei ole ilmselgelt see, mida otsib teadlane, kes peab oma järeldusi tegema mittetäieliku informatsiooni tingimustes.

Tõenäosusteooria on matemaatika haruna deduktiiivne aksiomaatiline süsteem, aga mõtlemise mudelina kasutatakse seda hoopiski induktiivsel moel. See tähendab, et me püüame piiratud andmete põhjal jõuda ebakindlatele järeldustele, aga seejuures seda ebakindlust tõenäosustega kvantifitseerides. Uusi andmeid lisades saame me oma episteemilise ebakindluse (“episteemiline ebakindlus” tähendab, et segadus asub meie peas, mitte maailma ülesehituses) määra muuta, aga ainus viis saavutada tõsikindlust (ja monotoonilisust), on tuues arvutusse sisse null- ja ühiktõenäosused. Seega on tõenäosusteooriat mõtlemise mudelina rakendades teaduslikus praktikas üsna võimatu jõuda tõsikindlatele järeldustele. Ja inimesed, kes teaduses opereerivad lausearvutusliku loogikaga (ja seega ei mõtle tõenäosuslikult), eeldavad vaikimisi, et nende jaoks on teadus matemaatikat meenutav tõsikindel süsteem, mis oma sisendites (katseskeemid, andmed, nende analüüs) ei sisalda ebakindlust. Samas, ka meie, kes me kasutame tõenäosusteooriat, peame eeldama, et see on vaid mõtlemise mudel, mitte teaduslik mõtlemine ise oma ehedal kujul. Kohe, kui keegi mõtleb välja parema mudeli, hakkame kõik kasutama seda. Aga senikaua peame õppima tõenäosusteooriat ja selle praktilist edasiarendust, mida kutsume Bayesi statistikaks.

Chapter 11

Järeldav statistika

Kui EDA määrab graafiliste meetoditega andmete kvaliteeti ja püstitab uusi hüpoteese, siis järel dav statistika püüab formaalsete arvutuste abil vastata kahele lihtsale küsimusele: 1. mis võiks olla kõige usutavam parameetrväärtus? ja 2. kui suur ebakindlus seda hinnangut ümbritseb? Kuna andmed tulevad meile lõpliku suurusega valimina koos mõõtmisveaga ja bioloogilise varieeruvusega, on ebakindlus hinnagusse sisse ehitatud. Hea protseduur kvantititseerib selle ebakindluse ausalt ja täpselt – siin ei ole eesmärk niivõrd mitte ebakindlust vähendada (seda teeme eelkõige katse planeerimise tasemel), vaid seda kirjeldada. Järeldav statistika püüab, kasutades algoritme ja mudelite, teha andmete põhjal järel dusi looduse kohta.

Ebakindluse allikad on mõõtmisviga, bioloogiline varieeruvus, mudeli viga (matemaatiline jaotusfunktsioon ei vasta looduslikeks toimuvateks), algoritmi viga (algoritm ei tee seda, mida kasutaja tahab) ja süstemaatiline viga (juhtub, kui te saate valesti aru oma katsesüsteemist, harrastate teaduslikku pettust või teete kõike muud, mis kallutab teie andmeid). Süstemaatilist viga ei saa kunagi välistada – see on loogiline paratamus, mis tuleneb asjaolust, et andmeid on alati vähem kui on võimalikke süstemaatilise vea allikaid (ja sest lisaks ei ole meil kunagi täielikku nimekirja võimalikest vea allikatest), mille vahel saab loogiliselt vahet teha ainult neidsamaseid andmeid kasutades. Seega on vea allikad, millest igaüks on vaadeldav eraldi seisva alternatiivse teadusliku hüpoteesina meie põhihüpoteesile, andmete poolt loogiliselt alamääratud, mis tähendab et kindel teadmine teaduses on loogiliselt võimatu. Seega ulatuvad teaduslikud hüpoteesid alati teistpoolte andmete, ehk need ütlevad rohkem, kui oleks võimalik puhtalt andmete põhjal õigustada, ja sisaldavad usukomponenti. See on üks põhjas, miks me sõandame riskida mudelite kasutamisega oma hüpoteeside kinnitamisel.

Sellisel tegevusel on mõtet ainult siis, kui ühest küljest andmed peegeldavad tegelikkust ja teisest küljest tegelikkus hõlmab enamat, kui lihtsalt meie andmeid. Kui andmed = tegelikkus, siis pole mõtet keerulisi mudeliteid kasutada – piisab lihtsast andmete kirjeldusest. Ja kui andmetel pole midagi ühist tegelikkusega, siis on need lihtsalt ebarelevatsed. Seega on järeldava statistika abil tehtud järeldused alati rohkem või vähem ebamäärasel ning meil on vaja meetodit selle ebamäärasuse mõõtmiseks. Selle meetodi annab tõenäosusteooria.

Järeldav statistika on tõenäosusteooria käepikendus

See õpik õpetab Bayesi statistikat, mis põhineb tõenäosusteorial. Tänu sellele moodustab Bayesi statistika sidusa terviku, mille abil saab teha kõike seda, mida saab teha tõenäosusteooria abil. Bayesi statistika põhineb Bayesi teoreemil, mis on triviaalne tulelus tõenäosusteooria aksioomidest. Tänu Cox-i teoreemile (1961) teame, et klassikaline lausearvutuslik loogika on tõenäosusteooria erijuht ning, et Bayesi teoreem on teoreetiliselt parim viis tõenäosustega töötamiseks. Seega, kui te olete kindel oma väidete töesuses või vääruses, siis on klassikaline loogika parim viis nendega opereerida; aga kui te ei saa oma järeldustes pärts kindel olla, siis on teoreetiliselt parim lahendus tõenäosusteooria ja Bayesi teoreem.

Lisaks kooskõlalisusele tõenäosusteooriaga eristab Bayesi statistikat klassikalisest sageduslikust statistikast kaks põhilist asjaolu: Bayesis kasutatakse episteemilist tõenäosuse tõlgendust, mille kohaselt tõenäosus mõõdab usu määra hüpoteesi kehtimisse, ja teiseks, iga Bayesi arvutus sisaldab lisaks andmetele meie neist andmetest sõltumatut hinnangut hüpoteesi kehtimise tõenäosusele (see hinnang on formaliseeritud nn eeljaotuses ehk eeltõenäosuses ehk prioris). Ortodoksne nn sageduslik statistika kasutab seevastu nn objektiivset ehk sageduslikku tõenäosuse tõlgendust ja ei kasuta eeltõenäosusi oma arvutustes. Põhjalikumat seletust vt allpool ja lisa 1.

Tõenäosusteooria on aksiomaatiline süsteem, mille abil saame omistada numbriline väärtsuse meie usu määrale mingisse hüpoteesi. Näiteks, kui me planeerime katset, kus me viskame kulli ja kirja ja teeme seda kaks korda, siis saame arvutada, millise tõenäosusega võime oodata katse tulemuseks kaht kirja. Aga seda tingimusel, et me võtame omaks mõned eeldused – näiteks et münt on aus ja et need kaks viset on üksteisest sõltumatud.

Sellel katsel on 4 võimalikku tulemust: H-H, H-T, T-H, T-T (H - kull, T - kiri). Tõenäosus saada 2-1 mündiviskel 2 kirja, $P(2 \text{ kirja}) = 1/4$, $P(0 \text{ kirja}) = 1/4$ ja $P(1 \text{ kiri}) = 2/4 = 1/2$. Sellega oleme andnud oma katseplaanile täieliku tõenäosusliku kirjelduse (pane tähele, et $1/4 + 1/4 + 1/2 = 1$). Ükskõik kui keeruline on teie katseplaan, põhimõtteliselt käib selle tõenäosusteoreetiline analüüs samamoodi. Tõenäosusteooria loomus seisneb kõikide võimalike sündmuste üleslugemises ja

erinevat tüüpi sündmuste suhteliste sageduste arvutamises – ning senikaua, kui me seda nüri järjekindlusega teeme, on vastus, mille me saame, tõsikindel.

Ehkki Bayesi statistika põhineb tõenäosusteoorial ja on sellega kooskõlas, ei ole see sama asi, mis tõenäosusteooria. Statistikas pööratakse tõenäosusteooreetiline ülesanne pea peale ja küsитakse nii: kui me saime 2-l mündiviskel 2 kirja, siis millise tõenäosusega on münt aus (tasakaalus)? Erinevus tõenäosusteooreetilise ja statistilise lähenemise vahel seisneb selles, et kui tõenäosusteoorias me eeldame, et teame, kuidas süsteem on üles ehitatud, ja ennustame sellest lähtuvalt võimalike (hüpoteetiliste) andmete tõenäosusi, siis statistikas me kontrollime neid eeldusi päriselt olemasolevate andmete põhjal. Seega annab tõenäosusteooria matemaatiliselt tõsikindlaid vastuseid ideaalmaailmade kohta, samas kui statistika püüab andmete põhjal teha järeldusi päris maailma kohta. Selleks kasutame Bayesi teoreemi (vt allpool).

Tõenäosusteooria määrab kõikide võimalike sündmuste esinemise tõenäosused, eeldades, et hüpotees H kehtib (H on siin lihtsalt teine nimi “eeldusele”).

Statistika hindab H -i kehtimise tõenäosuse lähtuvalt kogutud andmetest, matemaatilistest mudelitest ning taustateadmistest.

Tõenäosusteooria aksioomid on tuletatavad järgmistest eeldustest:

- hüpoteesi usutavuse määra saab kirjeldada reaalarvudega 0 ja 1 vahel
- ratsionaalne mõtlemine vastab kvalitatiivselt tervele mõistusele: tõendusmaterjal hüpoteesi toetuseks tõstab selle hüpoteesi usutavust.
- mõtlemine peab olema konsistentne: kui me saame järeldusi teha rohkem kui ühel viisil, peame lõpuks ikkagi alati samale lõppjäreldusele jõudma
- kogu kättesaadav relevantne informatsioon tuleb järelduste tegemisel arvesse võtta (totaalse informatsiooni printsipi)
- ekvivalentsed teadmised on representeeritud ekvivalentsete numbritega.

Tõenäosusteooria aksioomid, mida on neli tükki, ütlevad tõlkes inimkeelde,

- (1) et iga sündmuse tõenäosus on suurem või võrdne nulliga - $P(A) \geq 0$,
- (2) et loogiliselt paratamatu sündmuse tõenäosus on üks - $P(\Omega) = 1$. Ω on *sample space*, mis koosneb üksteist välistavatest ja ammendavatest hüpoteesidest. Nende hüpoteeside tõenäosuste summa on 1, mis tähendab, et täpselt üks neist kehtib paratamatult.
- (3) et üksteist välistavate sündmuse puhul võrdub tõenäosus, et toimub üks või teine sündmus, nende sündmuste tõenäosuste summaga, ehk $P(A \vee B) = P(A) + P(B)$.
- (4) et sündmuse A tõenäosus, juhul kui me eeldame sündmuse B kehtimist, võrdub nende kahe sündmuse koosesinemise tõenäosuse jagatisega sündmuse

B tõenäosusest, ehk $P(A | B) = \frac{P(A \wedge B)}{P(B)}$.

Sümbolite tähindused:

- $A \wedge B$ tähindab “ A ja B ”,
- $A \vee B$ tähindab “ A või B ”,
- $\neg A$ tähindab mitte- A , ehk $A == \text{FALSE}$.
- $P(A | B)$ on tinglik tõenäosus, mida tuleks lugeda: “ A tõenäosus tingimusel, et kehtib B ”. Sellega me ei väida, et B päriselt kehtib, vaid küsime: “Kui peaks juhtuma, et B kehtib, milline oleks sellisel juhul A tõenäosus?”. Pane tähele, et $P(vihm | pilves ilm)$ ei ole numbriliselt sama, mis $P(pilves ilm | vihm)$.

Need aksioomid, mis oma matemaatilises vormis postuleeriti Andrei Kolmogorovi poolt ca 1933, peaksid olema iseenesestmõistetavad ja ainult neist on toletatud kogu tõenäosusteooria. Tõenäosusteooria on matemaatika haru, mis tähindab, et sümbolitel P , A , B , jms ei ole selles muud fikseeritud tähindust, kui et need käituvad vastavalt tõenäosusteooria aksioomidele ja neist dedutseeritud teoreemidele. Nendes piirides võime anda neile sümbolitele anda ükskõik millise tähinduse, mis siis seob matemaatilise struktuuri pärис maailmaga, mis ei ole oma loomuselt matemaatiline formalism. Näiteks $P(A | B)$ võib tähindada “hüpoteesi A tõenäosust tingimusel, et meil on andmed B ”, aga sama hästi ka “andmete A tõenäosust tingimusel, et kehtib hüpotees B ”, või ka midagi muud. $P(A)$ võib meie jaoks tähistada “hüpoteesi tõenäosust”, “andmete tõenäosust”, “tõendusmaterjali tõenäosust” ja “sündmuse tõenäosust”, aga ka “homse vihma tõenäosust” või “tõenäosust, et parameetri väärus > 2 ”).

$A \vee \neg A$ tähistab loogiliselt paratamatut ehk loogiliselt tõest propositiooni ja $A \wedge \neg A$ tähistab loogiliselt vastuolulist ehk loogiliselt väärat propositiooni (propositioon on defineeritud lausena, millele saab omistada tõeväärtuse). Loogiliselt väära propositiooni tõenäosus on null ja loogiliselt tõese propositiooni tõenäosus on üks ($P(\neg A) = 1 - P(A)$).

Kui tõenäosused on 0 või 1, siis taandub tõenäosusteooria matemaatiliselt oma erijuuhule, milleks on lausearvutuslik loogika. Lausearvutusel on huvitav omadus, monotoonilisus, mille kohaselt kui juba on saavutatud loogiliselt validne tulemus, siis uute andmete lisandumisel ei saa me seda muuta. Seestast tõenäosusteoorias ja statistikas muudavad uued andmed hüpoteesi kehtimise tõenäosust. Selles mõttes ei saa tõenäosuslik teadus kunagi valmis ja kui inimene on 100% veendunud mingi hüpoteesi/sündmuse tõesuses või vääruses, siis seisavad tema uskumused väljaspool teadust selles mõttes, et neid ei ole võimalik teaduslike argumentidega mõjutada.

Formaalsed tuletised tõenäosusteooria aksioomidest

Me anname siin 9 tuletust ilma tõestuskäikudeta, mis on aga lihtsad. Siin võib A ja B vaadelda erinevate sündmustena või hüpoteesidena. Me eeldame, et kummagi hüpoteesi tõenäosus > 0 .

Tõenäosusteooria põhituletised:

5. $0 \leq P(A) \leq 1$ - tõenäosused jäävad 0 ja 1 vahele
6. $P(\neg A) = 1 - P(A)$, üksteist välisavate ammendavate hüpoteeside tõenäosused summeeruvad ühiktõenäosusele.
7. $P(A \ \& \ B) \leq P(A); P(B) \leq P(A \vee B)$
8. Kui B tuleneb dedukiivselt A-st ja $P(A) > 0$, siis $P(B | A) = 1$ ja $P(\neg B | A) = 0$. Siit tuleeb, et kui tõendusmaterjal e tuleneb dedukiivselt hüpoteesist H (H ennustab e-d) ja kui $P(H) > 0$ ning $P(e) < 1$, siis $P(H | e) > P(H)$, ehk e tõstab H tõenäosust.
9. Kui B tuleneb dedukiivselt A-st $[A \vee (B \wedge \neg A) \Leftrightarrow B]$, siis $P(B) \leq P(A)$.
10. Loogiliselt ekvivalentsed propositioonid/hüpoteesid on sama tõenäosusega – kui $A \Leftrightarrow B$, siis $P(A) = P(B)$
11. Üksteist välisavate propositioonide korral $P(A_1 \vee A_2 \vee \dots \vee A_n) = P(A_1) + P(A_2) + \dots + P(A_n)$ – see on *finitse additiivsuse printsipi*.
12. Defintsioon: A ja B on üksteisest sõltumatud siis ja ainult siis kui $P(A | B) = P(A)$
13. Kui A ja B on üksteisest sõltumatud, siis

$$P(A \wedge B) = P(A)P(B)$$

14. Kui A ja B ei ole üksteisest sõltumatud, siis

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

ja kolmele sündmusele:

$$\begin{aligned} P(A \vee B \vee C) &= P(A) + P(B) + P(C) - \\ &P(A \wedge B) - P(B \wedge C) - P(A \wedge C) + \\ &P(A \wedge B \wedge C) \end{aligned}$$

15. Totaalne ehk marginaalne tõenäosus:

$$P(A) = P(A | B)P(B) + P(A | \neg B)P(\neg B)$$

ehk

$$P(A) = P(A | B_1)P(B_1) + P(A | B_2)P(B_2) + \dots$$

üksteist välisavatele B-dele.

16. Bayesi teoreem:

$$P(A | B) = \frac{P(A)P(B | A)}{P(B)}$$

kus vastavalt 15. punktile

$$P(B) = P(A)P(B | A) + P(\neg A)P(B | \neg A)$$

või

$$P(B) = P(A_1)P(B | A_1) + P(A_2)P(B | A_2) + \dots$$

Bayesi teoreemi kasutatakse määramaks hüpoteesi tõenäosuse pärist uute faktide (andmete) lisandumist olemasolevatele teadmistele. Selleks peab hüpoteesiruum olema jagatud vähemalt kaheks ammendavaks ja üksteist välistavaks hüpoteesiks. Kui A on H_1 ning mitte- A on ammendav ja välistav H_2 ja B tähistab andmeid (data), saame Bayesi teoreemi ümber kirjutada

$$P(H_1 | data) = \frac{P(H_1)P(data | H_1)}{P(H_1)P(data | H_1) + P(H_2)P(data | H_2)}$$

$P(H_1 | data)$ on H_1 kehtimise tõenäosus meie andmete korral – ehk posteerior,

$P(H_1)$ on H_1 kehtimise eelnev, ehk meie andmetest sõltumatu, tõenäosus – ehk prior,

$P(data | H_1)$ on andmete esinemise tõenäosus tingimusel, et H_1 kehtib – ehk tõepära.

Jagamistehel tehakse ainult selle pärist, et normaliseerida 1-le kõikide hüpoteeside tõenäosuste summa meie andmete korral ja seega viia posteerior vastavusse tõenäosusteooria aksioomidega — kui meil on i ammendavat üksteist välistavat hüpoteesi, siis murrujoone alla läheb $\sum P(data | H_i)P(H_i) = 1$.

Bayesi teoreem on triviaalne tuletus tõenäosusteooria aksioomidest, milles pole midagi maagilist. See ei ole automaatne meetod, mis tagaks inimkonna teadmiste kasvu, vaid lihtsalt parim võimalik viis andmemudeli ja taustateadmiste mudeli ühendamiseks ja normaliseerimiseks tinglikuks tõenäosuseks (hüpoteesi tõenäosus meie andmete ja taustateadmiste korral). Edasi sõltub kõik mudelite, andmete ja taustateadmiste kvaliteedist.

Näited tõenäosusteooria tuletiste rakendamisest

Järgnevatel näidetel on ühist kaks asja: need on matemaatiliselt triviaalselt lihtsad, aga intuitiivselt lootusetult keerulised. Kõigi nende puhul on inimestel tugev intuitsioon, mis on vale – ja tõenäosusteooria tundmine ei anna meile paremat intuitsiooni. Seega, ainus, mis üle jääb, on iga probleemi taandamine tõenäosusteooria valemitele ja selle tuimalt läbi arvutamine.

7. Punkt Linda on 31 aastane, vallaline, sõnakas ja väga nutikas. Ta õppis ülikoolis filosoofiat ja muretses sel ajal sügavalt diskrimineerimise ja sotsiaalse õigluse pärast ning osales tuumarelva vastastel meeleväldustel. Kumb on tõenäolisem? Linda on pangateller. Või Linda on pangateller, kes osaleb feministlikus liikumises. Kuigi enamus vastajatest eelistab 2. varianti, on see sõna otses mõttes loogikavastane.

13. Punkt Kui me viskame täringut 3 korda, kui suure tõenäosusega saame vähemalt ühe kuue? Naiivselt võiks arvata, et see tõenäosus on 50%. Kuid rakendades tõenäosusteooriat saame teistsuguse vastuse. Lihtsuse huvides defineerime küsimuse ümber: kui suure tõenäosusega ei saa me 3-l viskel ühtegei kuute? Vastus: kui igal viskel on 0 kuue tõenäosus $5/6$, siis $(5/6)(5/6)(5/6) = 0.58$ ja $1 - 0.58 = 0.42$, mis tähendab, et vähemalt 1 kuue (või ükskõik mis numbri ühest kuueni) saame 42% tõenäosusega. Teine näide (NYT 03-12-2017): te ostate maja Texases Hustonis, millele müüja annab garantii, et üleujutuse tõenäosus on 1% aastas. Seadus nimetab seda näidikut "100 aasta suurvee-tasemeks". 1% näidu puhul ei pea te seaduse järgi ostma üleujutusekindlustust. Kui suure tõenäosusega tabab teie maja üleujutus pangalaenu perioodi välitel (30 aastat)? Vastus: $1 - (99/100)^{30} = 0.26$.

14. Punkt Kui tõenäosus, et homme sajab pussnuge on 0.1 ja et ülehomme sajab pussnuge on 0.1, siis millise tõenäosusega sajab vähemalt ühel neist päevadest? Eeldades sündmuste sõltumatust:

$$\begin{aligned} P(\text{homme sajab} \vee \text{ülehomme sajab}) &= \\ 0.1 + 0.1 - 0.1 \times 0.1 &= 0.19 \end{aligned}$$

Kui me aga teame, et sadu erinevatel päevadel on korreleeritud näiteks nii:

$$\begin{aligned} P(\text{ülehomme sajab} \mid \text{homme sajab}) &= 0.2 \\ P(\text{ülehomme sajab} \mid \neg\text{homme sajab}) &= 0.15 \end{aligned}$$

siis

$$\begin{aligned} P(\text{homme sajab} \vee \text{ülehomme sajab}) &= \\ P(\text{homme sajab}) + P(\text{ülehomme sajab}) - & \\ P(\text{homme sajab} \& \text{ülehomme sajab}) & \end{aligned}$$

Nüüd peame arvutama $P(\text{ülehomme sajab})$, kasutades 15. punkti (marginaliseerimist), misjärel saame valemi

$$P(A \vee B) = P(A) + P(B) - P(A)P(B \mid A)$$

Kui vihm on korreleeritud, siis väheneb tõenäosus, et sajab vähemalt ühel päeval.

4. Punkt Meil on kolm pannkooki, millest esimesel on mõlemad külged moosised, teisel on üks külj moosine ja kolmandal pole üldse moosi. Juhtus nii, et meile pandi taldrikule pannkook, mille pealmine külj on moosine. Millise tõenäosusega on moosine ka selle pannkoogi alumine külj? NB! Vastus ei ole 50%. Lahendus: Kui A - moos all, B - moos üleval, siis vastavalt 4. aksioomile

$$P(\text{moos all} \mid \text{moos üleval}) = \frac{P(\text{moos all} \wedge \text{moos üleval})}{P(\text{moos all})}$$

Tõenäosus, et moos on all ja üleval on $1/3$ (me teame, et 1 pannkook 3st on mõlemalt küljelt moosine) ja tõenäosus, et moos on all, on keskmene kolmest tõenäosusest, millega me kolmel pannkoogil moosise külje saame: $\text{mean}(\text{c}(1, 0.5, 0)) = 1/2$. Seega, vastus on $(1/3)/(1/2) = 2/3$. Kui me saame moosise ülemise külje, siis on tõenäosus $2/3$, et ka all on moos!

15. Punkt Kui A tähistab sündmust “ma sooritan eksami edukalt” ja B tähistab sündmust “ma õpin eksamiks”, ning meil on dihhotoomne valik: õpin / ei õpi, siis

$$\begin{aligned} P(\text{hea hinne}) &= P(\text{õpin})P(\text{hea hinne} \mid \text{õpin}) + \\ &P(\text{ei õpi})P(\text{hea hinne} \mid \text{ei õpi}) \end{aligned}$$

Ehk sõnadega kirjutatult: Hea hinde tõenäosus võrdub korrutisega kahest tõenäosustest – tõenäosus, et ma eksamiks õpin, ja tõenäosus, et ma saan hea hinde siis kui ma õpin –, millele tuleb liita teine korrus kahest tõenäosustest – tõenäosus, et ma ei õpi, ja tõenäosus, et ma saan hea hinde ka ilma õppimata. Siit saad ise enda jaoks välja arvutada ennustuse, millise tõenäosusega just sina selle kursuse edukalt läbid.

16. Punkt Bayesi teoreemi rakendamine diskreetsetele hüpoteesidele: Oletame, et 45 aastane naine saab rinnavähi sõeluuringus mammograafias positiivse tulemuse. Millise tõenäosusega on tal rinnavähk? Kõigepealt jagame hüpoteesiruumi kahe diskreetse hüpoteesi vahel: H_1 - vähk ja H_2 - mitte vähk. Edasi omistame numbrilised vääritud jäägmistele parameetritele:

1. H_1 tõepära, ehk tõenäosus saada positiivne mammogramm juhul, kui patsiendil on rinnavähk (testi sensitiivsus): $P(+ \mid H_1) = 0.9$
2. H_2 tõepära, ehk tõenäosus saada positiivne mammogramm juhul, kui patsiendil ei ole rinnavähki (1 - testi spetsiifilus): $P(+ \mid H_2) = 0.08$. Pane tähele, et $0.9 + 0.08$ ei võrdu ühega, mis tähendab, et tõepära pole tõenäosusteooria mõttes päris tõenäosus.
3. Eelnev tõenäosus, et patsiendil on rinnavähk $P(H_1) = 0.01$ (see on rinnavähi sagekus 45 a naiste populatsioonis; kui me teame patsiendi genoomi järjestust või rinnavähijuhete tema lähisugulastel, võib $P(H_1)$ tulla väga erinev).
4. $P(H_2) = 1 - P(H_1) = 0.99$

Nüüd arvutame posterioorse tõenäosuse $P(H_1 \mid +)$

```
likelihood_H1 <- 0.9
likelihood_H2 <- 0.08
prior_H1 <- 0.01
prior_H2 <- 1 - prior_H1
posterior1 <- likelihood_H1*prior_H1/
  (likelihood_H1*prior_H1 +
   likelihood_H2*prior_H2)
```

```
posterior1
#> [1] 0.102
```

Nagu näha, postiivne tulemus rinnavähi sõeluuringus annab 10% töenäosuse, et teil on vähk (ja 90% töenäosuse, et olete terve). Selle mudeli parameetrväärtused vastavad enam-vähem tegelikele mammograafia veasagedustele ja tegelikule populatsiooni vähisagedusele.

Mis juhtub, kui me teeme positiivsele patsiendile kordustesti? Nüüd on esimese testi posteerior meile prioriks, sest see kajastab definitsiooni järgi kogu teadmist, mis meil selle patsiendi vähiseisundist on (muidugi eeldusel, et me esimese mudeli kohusetundlikult koostasime).

```
likelihood_H1 <- 0.9
likelihood_H2 <- 0.08
prior_H1 <- posterior1
prior_H2 <- 1 - prior_H1
posterior2 <- likelihood_H1*prior_H1 /
  (likelihood_H1*prior_H1 +
   likelihood_H2*prior_H2)
posterior2
#> [1] 0.561
```

Patsiendile võib pärast kordustesti positiivset tulemust öelda, et ta on 44% töenäosusega vähivaba. Eelduseks on, et me ei tea midagi selle patsiendi geenetikast ega keskkonnast põhjustatud vastuvõtlikusest vähile ning, et testi ja kordustesti vead on üksteisest sõltumatud (mitte korreleeritud).

Bayesi teoreemi kasutamine pideva suuruse (näiteks keskväärtuse või standardhälbe) hindamiseks on põhimõtteliselt samasugune, ainult et nüüd on meil lõpmata suur arv hüpoteesi (iga teoreetiliselt võimalik parameetri väärthus on siin “hüpotees”), mis tähindab, et vastavalt Bayesi teoreemile on meil vaja ka lõpmata hulka töepärasid ja lõpmata hulka prioreid. Lõpmata hulk töepärasid ja prioreid tähindab lihtsalt, et me avaldame need kahe pideva funktsionina, misjärel saame neist kahest funktsionist arvutada kolmanda pideva funktsioni, posteeriori. Posteeriorist saab omakorda arvutada iga möeldava parameetrväärtuste vahemiku töenäosuse või usalduspiirid, milles mingi meie poolt etteantud töenäosusega paikneb parameetri tegelik väärthus (vt ptk 10). Ja posteeriorse funktsiononi tipp (mood) vastab kõige töenäolisemale parameetrväärtusele.

Mida kitsam on posteerior, seda kitsamad tulevad sellest arvutatud usalduspiirid. Seega peaksime püüdma panna oma mudelitesse parameetreid (statistikuid), mille posteeriorid tulevad võimalikult kitsad (vt allpool ptk “ajalooline vahepala” selle kohta, kuidas aritmeetiline keskmene on selline statistik).

Tõenäosuse episteemiline tõlgendus

Tõenäosus $P()$ ei ole matemaatiliselt midagi enamat, kui reaalarv, mis rahuldab Kolmogorovi aksioomide poolt seadud tingimusi. Seevastu tõenäosuse mõiste tõlgendus, mis rahuldaks teaduse vajadusi, on pigem filosoofiline kui matemaatiline probleem. Teisisõnu, tõenäosusteooria õpetab meid tõenäosustega matemaatiliselt ümber käima, kuid ei anna meile seost matemaatiliste tõenäosuste ja päris maailma vahel, ega ei ütle, mida tõenäosus teaduses tähendab. Kaasajal kasutatkse kahte põhilist tõenäosuse tõlgendust, episteemilist (e Bayesiaanlikku) ja objektiivset (e sageduslikku), millest me siin käsitleme esimest. Sagedusliku tõlgenduse kohta vt lisa 1.

Kuna kõik tõenäosuse tõlgendused alluvad samadele tõenäosusteooria aksioomidele, siis on kõik valiidsed tõenäosust sisaldavad argumendid tõlgitavad erinevate tõenäosuste tõlgendustele vahel. Seda on tähtis teada. Võtame näiteks olukorra, kus hüpotees h deduktiiivselt ennustab tõendusmaterjali e -d ja lisaks oleks e väga ebatõenäoline juhul, kui h ei kehti. Paraku, isegi kui katse tulemus on tõepoolest e , ei saa me ikkagi ilma h -i eeltõenäosust $P(h)$ kasutamata, öelda midagi h -i kehtimise tõenäosuse kohta meie andmete korral, sest Bayesi teoreem kehtib ühtmoodi kõikide tõenäosuse tõlgendustele korral. Aga just seda teevald pahatihi teadlased (ja harvem statistikud), kes kasutavad traditsionilist sageduslikku statistikat, mis ei sisalda formaalseid meetodeid eeltõenäosuste arvesse võtmiseks. Teisalt pole matemaatilises mõttes vahet, millist tõenäosuse tõlgendust te parjasagu kasutate -- see on teie teaduslik ja filosoofilise eelistus, ja te võite tõenäosust vastavalt vajadusele kasvõi iga päev erinevalt tõlgendada.

Bayesiaanlik statistika opereerib episteemilise tõenäosusega. See tähendab, et tõenäosus annab numbrilise mõõdu meie ebakindluse määrale mõne hüpoteesi ehk parametrväärtuse kehtimise kohta. Seega mõõdab tõenäosus meie teadmiste kindlust (või ebakindlust). Näiteks, kui arvutus näitab, et homme on vihma tõenäosus 60%, siis me oleme 60% kindlad, et homme tuleb vihma. Aga hoolimata sellest, mida me vihma kohta usume, homme kas sajab vihma või mitte, ja seega on homse vihma objektiivne tõenäosus meie akna taga 0 või 1 – ja mitte kunagi 0.6.

Tõenäosuse formaalne tõlgendus tuleb otse kihlveokontorist. Kui sa arvutasid, et vihma tõenäosus homme on 60%, siis see tähendab, et sa oled ratsionaalse olendina nõus maksma mitte rohkem kui 60 senti kihlveo eest, mis võidu korral toob sulle sisse 1 EUR – ehk 40 senti kasumit. Seega on “ausa kihlveo shansid” (fair betting odds) sinu jaoks 60:40 ehk 3:2 vihma kasuks, mis tähendab, et sa usud, et nende kihlveoshansidega oled sa enda jaoks tasakaalustanud riski nii võidu kui kaotuse korral ja usud, et pikas perspektiiviis jääd sa nii mängides nulli. Seega, ausa kihlveo shansid a:b annavad episteemilise tõenäosuse valemiga $a/(a + b)$, ja tõenäosusest a saab

sansid valemiga $b = 1 - a$.

Selles mõttes on Bayesi tõenäosus subjektiivne. Kui me teaksime täpselt, mis homme juhtub, siis ei oleks meil selliseid tõenäosusi vaja. Seega, kui te hoolimata kõigest, mida ma selle kohta eelpool kirjutanud olen, ikkagi usute, et teadus suudab tööstada väiteid maailma kohta samamoodi, nagu seda teeb matemaatika formaalsete struktuuride kohta, siis pääsete sellega statistika õppimisest ja kasutamisest. Aga kui te siiski arvutate Bayesi tõenäosusi, siis ei ütle need midagi selle kohta, kas maailm on tõenäosuslik või deterministlik. Inimesed, kes vajavad tõenäosusi maailma seisundite kirjeldamiseks, ei kasuta episteemilisi tõenäosusi, vaid sageduslikku tõlgendust.

Kui mõõdame pidevat suurust, näiteks inimeste pikkusi, siis saame arvutuse tagajärvel tõenäosused kõigi võimalike parameetrväärtuste kohta, ehk igale mõeldavale pikkuse väärtsusele. Kuna pideval suurusel on lõpmata hulk võimalikke väärtsusi, avaldame me sellised tõenäosused pideva tõenäosusfunktsioonina, e järeljaotusena e posteeriorina. Posteerior näeb sageli välja nagu normaaljaotus ja me võime selle põhjal arvutada, kui suur osa summaarest tõenäosusest, mis on 1, jäääb meid huvitavasse pikku vahemikku. Kui näiteks 67% posteriori pindalast jäääb pikku vahemikku 178 kuni 180 cm, siis me usume, et 0.67-se (67%-se) tõenäosusega asub tegelik keskmise pikkus kuskil selles vahemikus.

Tõenäosusteooriast tulenevad statistika põhiprintsiibid

1. statistilise analüüsi kvaliteet sõltub mudeli eeldustest & struktuurist. Kuna maailm ei koosne matemaatikast, teevad matemaatilised mudelid alati eeldusi maailma kohta, mis ei ole päris töesed ja mida ei saa tingimata empiiriliselt kontrollida. Mündiviske näites eeldasime, et mündivisked olid üksteisest sõltumatud. Kui me sellest eeldusest loobume, läheb meie mudel keerulisemaks, sest me peame mudelisse lisama teavet visetevahelise korrelatsiooni kohta. Aga see keerulisem mudel toob sisse uued eeldused. Üldiselt peaks mudeli struktuur kajastama katse struktuuri, mis kaasaegses statistikas tähendab sageli hierarhilisi mudeleid.
2. statistilise analüüsi täpsus sõltub andmete hulgast. Kui kahe mündiviske asemel teeksime kakskümmend, siis saaksime samade eelduste põhjal teha oluliselt väiksema ebakindluse määraga järeldusi mündi aususe kohta.
3. statistilise analüüsi kvaliteet sõltub andmete kvaliteedist. Kui münt on aus, aga me viskame seda ebauausalt, siis, mida rohkem arv kordi me seda teeme, seda tugevamalt usub teadusüldsus selle tagajärvel millessegi, mis pole tõsi.
4. statistilise analüüsi kvaliteet sõltub taustateadmiste kvaliteedist. Napid taustateadmised ei võimalda parandada andmete põhjal tehtud järeldusi juhul, kui andmed mingil põhjusel ei vasta tegelikkusele. Adekvaatsete taustateadmiste lisamine mudelisse aitab vältida mudelite üle-fittimist.

5. Järeldused ühe hüpoteesi kohta mõjutavad järeldusi ka kõikide alternatiivsete hüpoteeside kohta. Relevantsete hüpoteeside eiramine viib ekslikele järedustele kõigi teiste hüpoteeside kohta. Me ei saa põhimõtteliselt rääkida tõendusmaterjali tugevusest ühe hüpoteesi kontekstis — tõendusmaterjal on suhteline ja selle tugevust mõõdab tõepärade suhe $P(\text{andmed} | H_1)/P(\text{andmed} | H_2)$.

Andmed ei ole sama, mis tegelikkus

Nüüd, kus me saame aru tõenäosusteooriast, on aeg asuda statistika kallale. Me ei kasuta statistikat vabatahtlikult, vaid teeme seda ainult siis, kui usume kahte asja: ühest küljest, et meie andmed on piisavalt töetriuud, et nende põhjal saaks teha adekvaatseid oletusi päris maailma kohta. Ja teisest küljest, et meie andmed ei ole piisavalt töetriuud, et neid järedusi saaks teha lihtsalt ja intuitiivselt. Seega tasub alustada näitega sellest, kuidas andmed ja tegelikkus erinevad. Meie tööriistaks on siin simulatsioon. Simuleerimine on lahe, sest simulatsioonid elavad mudeli väikeses maailmas, kus me teame täpselt, mida me teeme ja mida on selle tagajärjel oodata. Simulatsioonidega saame hõlpsalt kontrollida, kas ja kuidas meie mudelid töötavad, aga ka genereerida olukordi (parameetrite väärusteks kombinatsioone), mida suures maailmas kunagi ette ei tule. Selles mõttes on mudelid korraga nii väiksemad kui suuremad kui päris maailm.

Alustuseks simuleerime juhuvalimi $n = 3$ lõpmata suurest normaaljaotusega populatsioonist, mille keskmene on 100 ja sd on 20. See on põhimõtteliselt sama simulatsioon, millise me tegime eelnevalt peatükis "Normaaljaotuse mudel väikestel valimitel". Jällegi, tähtis ei ole konkreetne juhuvalim, vaid valimi erinevus populatsioonist. Päriselus on korraliku juhuvalimi saamine tehniliselt raske ettevõtmine ja, mis veelgi olulisem, me ei tea kunagi, milline on populatsiooni tõeline jaotus, keskmene ja sd . Seega, elagu simulatsioon!

```
set.seed(1) # makes random number generation reproducible
Sample <- rnorm(n = 3, mean = 100, sd = 20)
Sample; mean(Sample); sd(Sample)
#> [1] 87.5 103.7 83.3
#> [1] 91.5
#> [1] 10.8
```

Nagu näha on meie konkreetse valimi keskmene 10% väiksem kui peaks ja valimi sd lausa kaks korda väiksem. Seega peegeldab meie valim halvasti populatsiooni — aga me teame seda ainult tänu sellele, et tegu on simulatsiooniga.

Kui juba simuleerida, siis roblina: tömbame ühe juhuvalimi asemel 10 000, arvutame seejärel 10 000 keskmist ja 10 000 sd -d ning vaatame nende statistikute jaotusi ja keskväärtusi. Simulatsioon on nagu tselluliit — see on nii odav, et igaüks võib seda endale lubada.

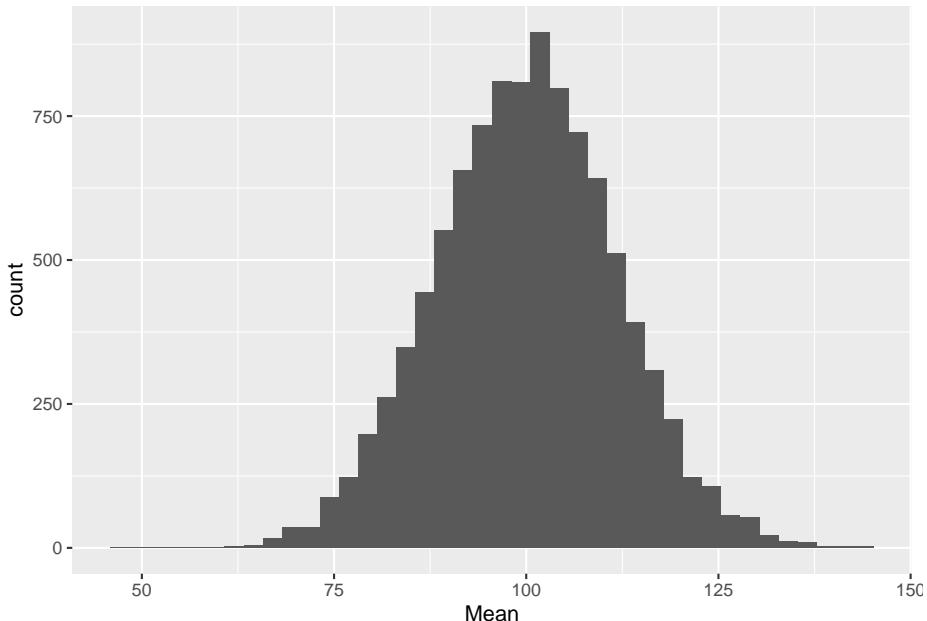


Figure 11.1: Keskmiste jaotus 10 000 valimist.

Meie lootus on, et kui meil on palju valimeid, millel kõigil on juhuslik viga, mis neid populatsiooni suhtes ühele või teisele poole kallutab, siis rohkem on valimeid, mis asuvad töelisele populatsioonile pigem lähemal kui kaugemal. Samuti, kui valimiviga on juhuslik, siis satub umbkaudu sama palju valimeid töelisest populatsiooniväärtusest ühele poole kui teisele poole ja vigade jaotus tuleb sümmeetrisline.

```
N <- 3
N_simulations <- 10000
df <- tibble(a = rnorm(N * N_simulations, 100, 20),
             b = rep(1:N_simulations, each = N))
Summary <- df %>%
  group_by(b) %>%
  summarise(Mean = mean(a), SD = sd(a))
Summary %>%
  ggplot(aes(Mean)) +
  geom_histogram(bins = 40)

mean(Summary$Mean)
#> [1] 100
mean(Summary$SD)
#> [1] 17.8
```

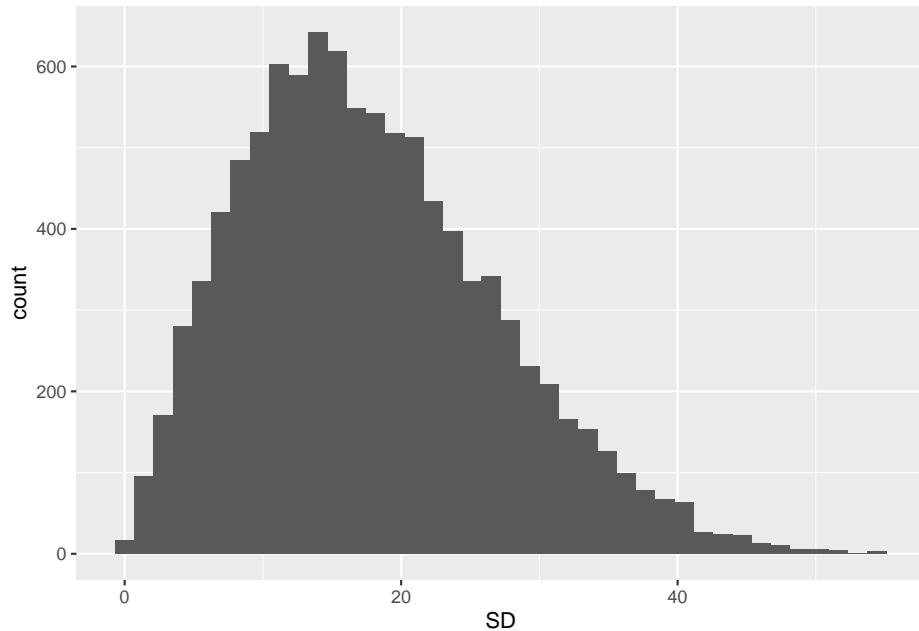


Figure 11.2: SD-de jaotus 10 000 valimist.

Oh-hooo. Paljude valimite keskmiste keskmise ennustab väga täpselt populatsiooni keskmist aga sd-de keskmise keskmise alahindab populatsiooni sd-d. Valem, millega sd-d arvutatakse, töötab lihtsalt kallutatult, kui n on väike (<10). Kes ei usu, kordab simulatsiooni valimiga, mille $N=30$.

Ja nüüd 10 000 SD keskväärtused:

```
Summary %>%
  ggplot(aes(SD)) +
  geom_histogram(bins = 40)

mode <- function(x, adjust = 1){
  x <- na.omit(x)
  dx <- density(x, adjust = adjust)
  dx$x[which.max(dx$y)]
}
mode(Summary$SD)
#> [1] 14.1
```

SD-de jaotus on ebasümmeetiline ja mood ehk kõige tõenäolisem valimi sd vääratus, mida võiksime oodata, on u 14, samal ajal kui populatsiooni sd = 20. Lisaks alahinnatud keskmisele sd-le on sd-de jaotusel paks saba, mis tagab, et teisest küljest pole ka vähetõenäoline, et meie valimi sd populatsiooni sd-d



Figure 11.3: Nii nagu parun Münchausen tõmbas ennast patsi pidi mülkast välja, genereeritakse bootstrappimisega algse valimi põhjal teststatistiku jaotus.

kõvasti üle hindab.

Arvutame, mitu % valimite sd-e keskmistest on > 25

```
mean(Summary$SD > 25)
#> [1] 0.211
```

Me saame $>20\%$ tõenäosusega pahasti ülehinnatud SD.

```
mean(Summary$SD < 15)
#> [1] 0.434
```

Ja me saame $>40\%$ tõenäosusega pahasti alahinnatud sd. Selline on väikeste valimite traagika.

Aga vähemalt populatsiooni keskmise saame me palju valimeid tõmmates ilusasti kätte — ka väga väikeste valimitega.

Kahjuks pole meil ei vahendeid ega kannatust loodusest 10 000 valimi kogumiseks. Enamasti on meil üksainus valim. Õnneks pole sellest väga hullu, sest meil on olemas analoogne meetod, mis töötab üsna hästi ka ühe valimiga. Me teeme lihtsalt ühest valimist mitu, mis meenutab pisut mittemillegist midagi tegemist, aga veidi üllatuslikult töötab selles kontekstis üsna hästi. Seda metoodikat kutsutakse *bootstrappimiseks* ja selle võttis esimesena kasutusele parun von Münchausen. Too jutukas parun nimelt suutis end soomülkast iseenda patsi pidi välja tõmmata (koos hobusega), mis ongi bootstrappimise põhimõte. (Inglise kultuuriruumis tõmbab bootstrappija ennast mülkast välja oma saapaserva pidi – siit ka meetodi nimi.) Statistika tõmbas oma saapaid pidi mülkast välja Brad Efron 1979. aastal.

Part II

OSA

Chapter 12

Bootstrap

```
library(rethinking)
library(bayesboot)
library(dplyr)
library(purrr)
library(ggplot2)
library(modelr)
```

Populatsioon on valimile sama, mis on valim bootstrappitud valimile.

Nüüd alustame ühestainsast empiirilisest valimist ja genereerime sellest 2000 virtuaalset valimit. Selleks tõmbame me oma valimist virtuaalselt 2000 uut juhuvalimit (bootstrap valimit), millest igaüks on sama suur kui algne valim. Saladus seisneb selles, et bootstrap valimate tömbamine käib asendusega, st iga empiirilise valimi element, mis bootstrap valimisse tömmatakse, pannakse kohe algsesse valimisse tagasi. Seega saab seda elementi kohe uesti samasse bootstrap valimisse tömmata (kui juhus nii tahab). Seega sisaldab tüüpiline bootstrap valim osasid algse valimi numbreid mitmes korduses ja teisi üldse mitte. Iga bootstrap valimi põhjal arvutatakse meid huvitav statistik (näiteks keskväärtus) ja kõik need 2000 bootstrapitud statistikut plotitakse samamoodi, nagu me ennist tegime valimitega lõpmata suurest populatsioonist. Ainsad erinevused on, et bootstrapis võrdub andmekogu suurus, millesse bootstrap valimeid tömmatakse, algse valimi suurusega ning, et iga bootstrapi valim on sama suur kui algne valim (sest meie poolt arvutatud statistiku varieeruvus, me tahame oma bootstrap valimiga tabada, sõltub valimi suurusest). Tüüpiliselt kasutatakse bootstrapitud statistikuid selleks, et arvutada usaldusintervall statistiku väärtsusele.

Bootstrap ei muuda meie hinnangut statistiku punktväärtusele. Ta annab hinnangu ebakindluse määrale, mida me oma valimi põhjal peaksime tundma selle punkthinnangu kohta.

Bootstrappimine on üldiselt väga hea meetod, mis sõltub väiksemast arvust eeldustest kui statistikas tavaks. Bootstrap ei eelda, et andmed on normaaljao-tusega või mõne muu matemaatiliselt lihtsa jaotusega. Tema põhiline eeldus on, et valim peegeldab populatsiooni – mis ei pruugi kehtida väikeste valimite korral ja kallutatud (mitte-juhuslike) valimite korral. Lisaks, tavaline bootstrap ei sobi hierarhiliste andmestruktuuride analüüsiks ega näiteks aegridade analüüsiks. Bootstrappida saab edukalt enamusi statistikuid, mida te võiksite elu jooksul arvutada, aga on siiski erandeid: näiteks valimi maksimum ja miinimumväärused.

Bootstrap empiirilisele valimile suurusega n töötab nii:

1. tõmba (asendusega) empiirilisest valimist B uut virtuaalset valimit (B bootstrap valimit), igaüks suurusega n .
2. arvuta keskmene, sed või mistahes muu statistik igale bootstrapi valimile. Tee seda B korda.
3. joonista oma statistiku väärustest histogramm või density plot

Nende andmete põhjal saab küsida palju toreidaid küsimusi — vt allpool.

Mis on USA presidentide keskmene pikkus? Meil on viimase 11 presidendi pikkused.

```
heights <- tibble(value = c(183, 192, 182, 183, 177, 185, 188, 188, 182, 185, 188))
B <- 1000 # the number of bootstrap samples

mean_value <- function(x) mean(as.data.frame(x)$value, na.rm = TRUE) # helper function
boot <- bootstrap(heights, B)
boot <- boot %>%
  mutate(Mean = map_dbl(strap, mean_value))

ggplot(boot, aes(Mean)) + geom_density()
```

ehk sama asi ilma bootsrtap funtsiooni kasutamata.

```
heights <- tibble(value = c(183, 192, 182, 183, 177, 185, 188, 188, 182, 185, 188))
n <- nrow(heights) #empirical sample size
B <- 1000 #nr of bootstrap samples
boot1 <- replicate(B, sample_n(heights, n, replace = TRUE))

df22 <- boot1 %>% #boot1 object is a list
  as.data.frame() %>% #convert this list into a data frame
  summarise_all(mean) %>% t() %>% as_tibble()

ggplot(df22, aes(V1)) + geom_histogram()
```

Mida selline keskväärtuste jaotus tähendab? Me võime seda vaadelda posterioorse tõenäosusjaotusena. Selle tõlgenduse kohaselt iseloomustab see jaotus täpselt meie usku presidentide keskmise pikkuse kohta, niipalju kui see usk põhineb

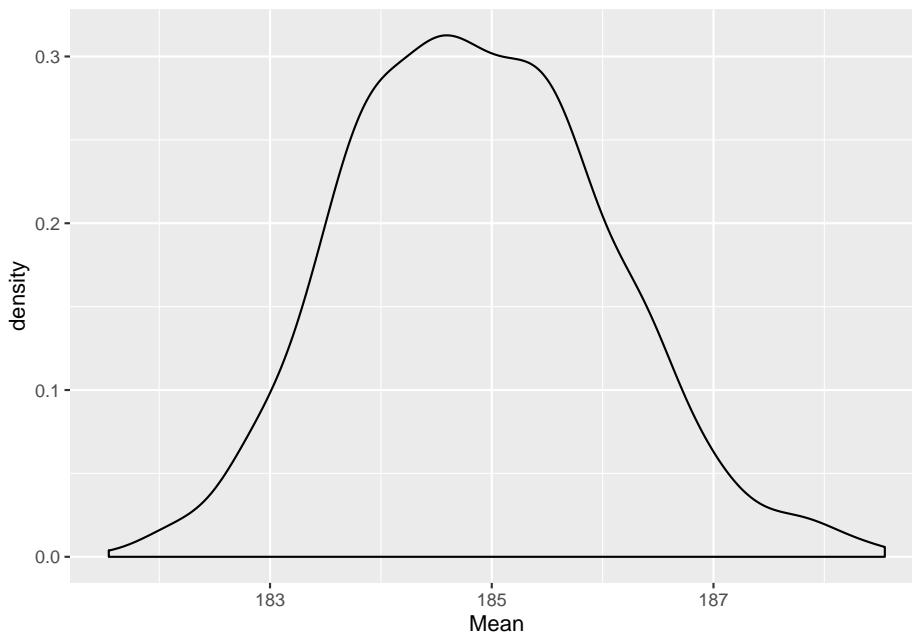


Figure 12.1: Bootstrapitud postseerior USA presidentide keskmisele pikkusele. Järgnevas koodis ütleme me kõigepealt, et $B = 2000$ (et me võtame 2000 bootstrap valimit) kasutades selleks broom paketi käsku `boot()`, millele on lihtne lisada `dplyr` funktsoon `summarise()`. Siiski peame seda tegema `dplyr::do()` abil. Pane tähele, et “tänu” `do()` kasutamisele peame me `summarise()` funktsoonis näitama punktiga koha, kuhu lähevad $\%>\%$ torust tulnud bootstrap valimid. Aga muidu on kõik tavapärane tidyverse.

bootstrappimises kasutatud andmetel. Senikaua, kui meil pole muud relevantset teavet, on köik, mida me usume teadvat USA presidentide keskmise pikkuse kohta, peidus selles jaotuses. Need pikkused, mille kohal jaotus on kõrgem, sisaldavad meie jaoks tõenäolisemalt tegelikku USA presidentide keskmist pikkust kui need pikkused, mille kohal posterioorne jaotus on madalam.

Kuidas selle jaotusega edasi töötada? See on lihtne: meil on 2000 arvu (2000 bootstrapitud statistiku väärust) ja me teeme nendega kõike seda, mida parasjagu tahame.

Näiteks me võime arvutada, millisesse pikkuste vahemikku jäääb 92% meie usust USA presidentide tõelise keskmise pikkuse kohta. See tähendab, et teades seda vahemikku peaksime olema valmis maksma mitte rohkem kui 92 senti piletit eest, mis juhul kui USA presidentide keskmise pikkus töesti jäääb sinna vahemikku, toob meile võidu suuruses 1 EUR (ja 8 senti kasumit). Selline kihlveokontor on täiesti respektaabel ja akadeemiline tõenäosuse tölgendus; see on paljude arvates lausa parim tölgendus, mis meil on.

Miks just 92% usaldusinterval? Vastus on, et miks mitte? Meil pole ühtegi universaalset põhjust eelistada üht usaldusvahemiku suurust teisele. Olgu meil usaldusinteval 90%, 92% või 95% — tölgendus on ikka sama. Nimelt, et me usume, et suure tõenäosusega jäääb tegelik keskväärtus meie poolt arvutatud vahemikku. Mudeli ja maailma erinevused tingivad niukuinii selle, et konkreetne number ei kandu mudelist otse üle pärismaailma. Eelnevalt mainitud kihlveokontor töötab mudeli maailmas, mitte teie kodulähedasel hipodroomil.

92% usaldusintervalli arvutamiseks on kaks meetodit, mis enamasti annavad vaid veidi erinevaid tulemusi.

1. HPDI — Highest Density Probability Interval — alustab jaotuse tipust (tippudest) ja katab 92% jaotuse kõrgema(te) osa(de) pindalast

```
HPDI(heights$value, prob = 0.92)
#> 10.92 0.92/
#> 177 192
```

2. PI — Probability Interval — alustab jaotuse servadest ja katab kummagist servast 4% jaotuse pindalast. PI 90%-le on sama, mis arvutada 5% ja 95% kvantiilid (jne).

```
PI(heights$value, prob = 0.90)
#> 5% 95%
#> 180 190
# quantile(heights$value, probs = c(0.05, 0.95)) teeb sama asja
```

HPDI on üldiselt parem mõõdik kui PI, aga teatud juhtudel on seda raskem arvutada. Kui HPDI ja PI tugevalt erinevad, on hea mõte piirduda jaotuse enda avaldamisega — jaotus ise sisaldab kogu informatsiooni, mis meil on oma statistiku vääruse kohta. Intervallid on lihtsalt summaarsed statistikud andmete kokkuvõtlikuks esitamiseks.

Kui suure tõenäosusega on USA presidentide keskmise pikkus suurem kui USA populatsiooni meeste keskmise pikkus (178.3 cm mediaan)?

```
mean(heights$value > 178.3)
#> [1] 0.909
```

Ligikaudu 100% tõenäosusega (valimis on 1 mees alla 182 cm, ja tema on 177 cm). Lühikesed jupatsid ei saa Ameerikamaal presidendiks!

Kuidas lahendada bootstrap, kui me tahame usaldusintervalle kahe ebavõrdse gruvi erinevusele? Näiteks kui meil on katsegrupis $N = 25$ ja kontrollgrupis $N = 20$, ja me tahame arvutada statistikut ES = katsegrupi keskmise - kontrollgrupi keskmise.

1. tõbma katsegrupist $N = 25$ bootstrapvalim
2. tõmba kontrollgrupist $N = 20$ bootsrapvalim
3. lahuta kontrollgrupi bootstrapvalimi mediaan katsegrupi omast (või aritmeetiline keskmise või ükskõik mis muu keskmise näitaja, mida hingihaldab)
4. korda punkte 1-3 B korda ja tööta edasi bootstrapjaptusega, nagu eespool näidatud.

12.1 Mõned tava-bootstrapi paketid

Professionaalid kasutavad boot paketti, mis on suhteliselt ebameeldiva süntaksiga, aga väga laialt rakendatav. Boot paketi peale on ehitatud tavainimesele hästi kasutatav pakett bootES (Kirby and Gelranc, 2013, Behav Res 45:905–927), mis teeb lihtsaks usalduspiiride leidmise erinevat tüüpi efekti suurustele, kaasa arvatur lihtsad hierarhilised ja ühefaktorilise ANOVA tüüpi katseskeemid. Nendes pakettides tasub üldjuhul kasutada meetodit nimega BCa (bias-corrected-and-accelerated) usalduspiiride arvutamiseks. See meetod piibub parandada bootstrap-valimite võimalikku kallutatust (esineb sedavõrd, kui bootstrap-jaotuse tipp ei ole samas kohas kui paljude päris-valimite pealt arvutatud statistikute jaotuse tipp) ja olukorda, kus statistiku väärtsuse varieeruvuse määrt sõltub statistiku väärtsusest. BCa edukaks arvutamiseks peab bootstrap valimite arv tuntavalalt ületama valimi suurust. Simulatsioonidega on näidatud, et BCa (ja teisi) usalduspiire saab mõistlikult arvutada valimitelt, mille suurus on > 15 . Sellest väiksemate valimite korral peate eeldama, et teie usaldusinettallid valemavad. Aegridade, kus esineb järjestikuste ajapunktide vahelisi sõltuvusi, tuleks kasutada nn block bootstrappi, mida implementerib näiteks boot::tsboot().

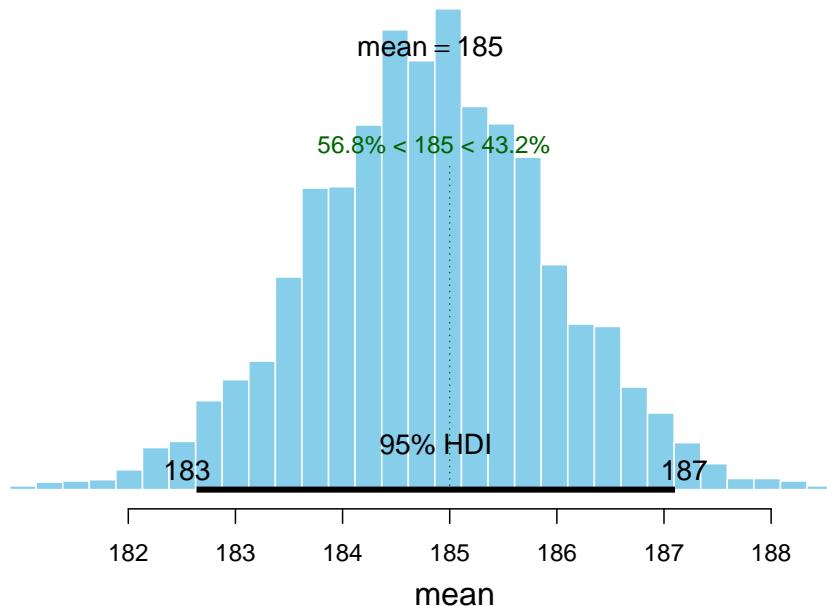


Figure 12.2: Bayesi bootstrapi posteerior USA presidentide keskmisele pikkusele.

Bayesi bootstrap

Kui klassikalise bootstrap meetodi pakkus välja B. Efron aastal 1979, siis selle Bayesi versioon avaldati D.B. Rubini poolt 1981. a. Bayesi versioon bootstrapist on implementeeritud “bayesboot” paketis funktsionis `bayesboot()`. Hea lihtsa seletuse Bayesi bootstrapi kohta saab siit <https://www.youtube.com/watch?v=WMAgzr99PK> ja lihtsa r koodi selle meetodi rakendamiseks saab siit <https://www.r-bloggers.com/simple-bayesian-bootstrap/>.

Lühidalt, erinevalt eelkirjeldatud tava-bootstrapist simuleeritakse Bayesi bootstrapis posterioorjaotused, näiteks arvutatakse kaalutud keskmise, kus ühtlastest jaotusest pärit kaalud on prioriks.

Näited sellest, kuidas kasutada bayesbooti standardhälbe, korrelatsioonikoeffitsiendi ja lineaarse mudeli koefitsientide usalduspiiride arvutamiseks leiate `?bayesboot` käsuga.

```
heights_bb <- bayesboot(heights$value, mean)
plot(heights_bb, compVal = 185)

HPDI(heights_bb$V1, prob = 0.95)
#> /0.95 0.95/
#>   183   187
```

Vaikimisi pannakse `bayesboot()` funktsioonis statistiku arvutamisel kaalud (prior) valimi indeksile, mis annab erineva tulemuse kui näiteks kaalutud keskmise arvutamisel, kus kaalud (prior) pannakse valimi väärustele.

Aritmeetilise keskmise Bayesi bootstrap vääritud kasutades kaalutud keskmise funktsiooni `weighted.mean` saab niimoodi:

```
heights_bb_w <- bayesboot(heights$value,
                           weighted.mean,
                           use.weights = TRUE)
```

Tõenäosus, et keskmene on suurem kui 182 cm

```
mean(heights_bb[, 1] > 182)
#> [1] 0.993
```

Kahe keskväärtuse erinevus (ES = keskmene1 - keskmene2):

```
set.seed(1)
## Simulate two random normal distributions with mean 0.
## True difference is 0.
dfr <- tibble(a = rnorm(10, 0, 1),
              b = rnorm(10, 0, 1),
              c = a - b)
dfr_bb <- bayesboot(dfr$c, weighted.mean, use.weights = TRUE )
plot(dfr_bb, compVal = 0)
```

BayesianFirstAid raamatukogu funktsioon `bayes.t.test()` annab kasutades t-jaotuse töepäramudelit üsna täpselt sama vastuse. See raamatukogu eeldab JAGS mcmc sämpleri installeerimist. Abi saab siit https://github.com/rasmusab/bayesian_first_aid ja siit <https://faculty.washington.edu/jmiyamoto/p548/installing.jags.pdf>.

Parameetriline bootstrap

Kui me arvame, et me teame, mis jaotusega on meie andmed, ja meil on suhteliselt vähe andmepunkte, võib olla mõistlik lisada bootstrapile andmete jaotuse mudel. Näiteks, meie USA presidentide pikkused võiksid olla umbkaudu normaaljaotusega (sest me teame, et USA meeste pikkused on seda). Seega fitime kõigepealt presidentide pikkusandmetega normaaljaotuse ja seejärel tõmbame bootstrap valimid sellest normaaljaotuse mudelist. Normaaljaotuse mudelil on 2 parameetrit: keskmene (mu) ja standardhälve (sigma), mida saame fittida valimandmete põhjal:

```
mu <- mean(heights$value)
sigma <- sd(heights$value)
N <- length(heights$value)
```

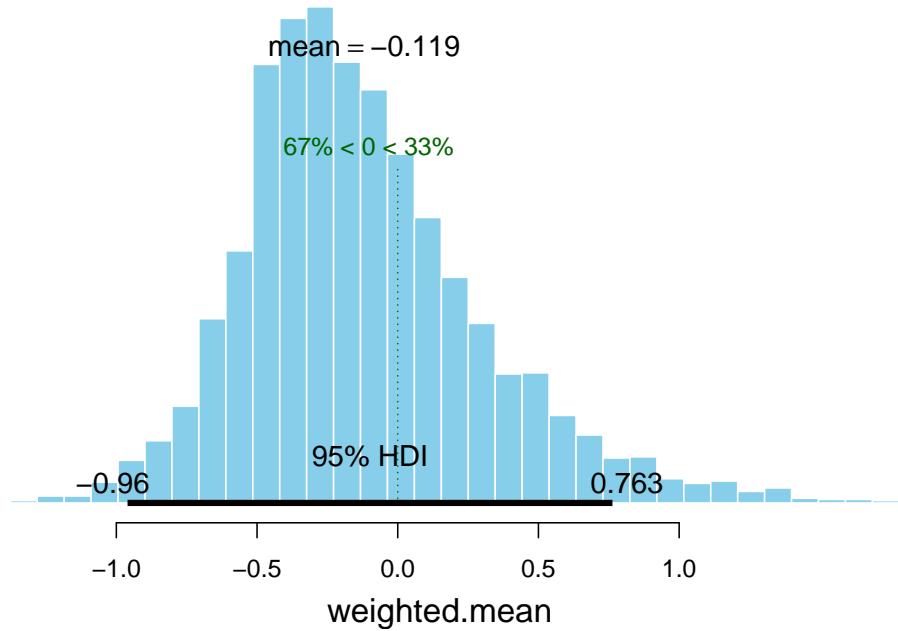


Figure 12.3: Bayesi bootstrap ES-le.

```
sample_means <- tibble(value = rnorm(N * 1000, mu, sigma),
                       indeks = rep(1:1000, each = N))

sample_means_sum <- sample_means %>%
  group_by(indeks) %>%
  summarise(Mean = mean(value))

ggplot(sample_means_sum, aes(x = Mean)) +
  geom_histogram(color = "white", bins = 20)

HPDI(sample_means_sum$Mean)
#> [1] 0.89 0.89
#> [2] 183   187
```

Üldiselt ei soovita me parameetrilist bootstrappi väga soojalt, sest täisbayesiaanlik alternatiiv, mida me kohe õppima asume, on sellest paindlikum.

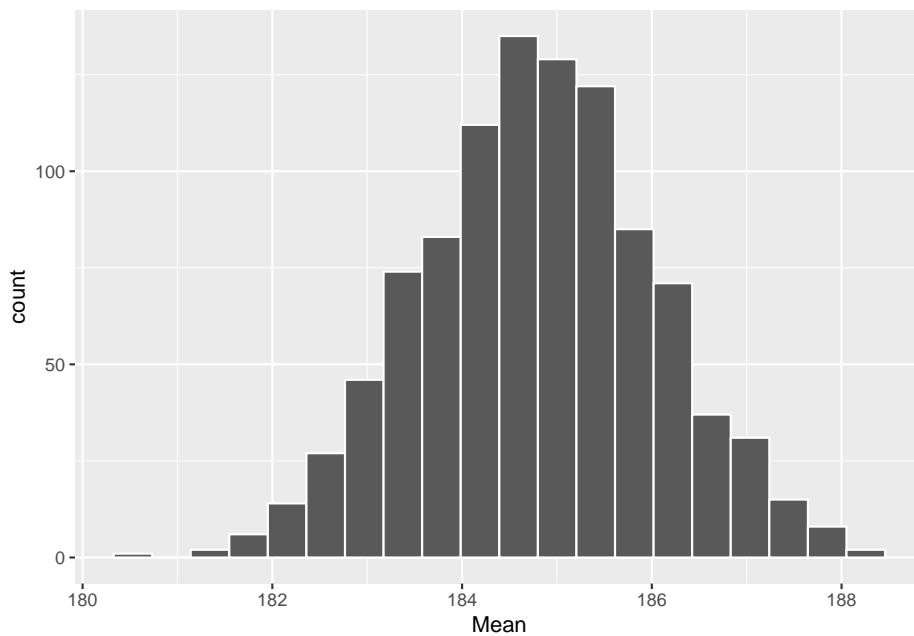


Figure 12.4: Parameetrilise bootstrapi posteerior USA presidentide keskmisele pikkusele.

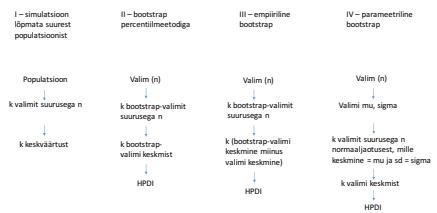


Figure 12.5: Bootstrappimise meetodid.

Bootstrappimine ei ole kogu tõde

Bootstrappimine on võimas ja väga laia kasutusalaga meetodite kogum. Sellel on siiski üks oluline piudus. Nimelt arvestab bootstrap ainult andmetega ja ignoreerib taustateadmisi (parameetriline bootstrap küll eeldab taustateadmisetele tuginevalt jaotusmudelit, kuid ignoreerib kogu muud taustateadmist). Miks on see probleem?

Mõtleme hetkeks sellele teadusliku meetodi osale, millel põhineb suuresti näiteks Darwini liikide tekkimise argument. See on nn *inference to the best explanation*, mille kohaselt on eelistatud see teoria, mis on parimas kooskõlas faktidega, ehk mille kehtimise korral on meie andmete esinemine kõige tõenäolisem. Kui mõni hüpotees omistab andmete esinemisele suure tõenäosuse, siis me ütleme tehnilises keeles, et see hüpotees on tõepärane (*has high likelihood*). Esmapilgul tundub see kõik igati mõistlik, kuid proovime lihtsat mõtteeksperimenti. Selles juhtub nii, et loteriil võidab peaauhinna meile tundmatu kodanik Franz K. Meil on selle fakti (ehk nende andmete) seletamiseks kaks teoriat: 1. Franz K. võit oli juhuslik (loterii oli aus ja keegi peab ju võitma) ja 2. Franz K. noorem õde võltsis loterii tulemusi oma venna kasuks. Teine teoria sobib andmetega palju paremini kui esimene (sest kuigi keegi peab võitma, Franz K. võiduvõimalus oli väga väike); aga ometi eelistab enamus mõistlikke inimesi esimest teoriat. Põhjas on selles, et meil pole iseenesest mingit alust arvata, et Franz K.-l üldse on noorem õde, või et see õde omaks ligipääsu loteriile. Kui me aga saame teada, et Franz K. noorem õde tõesti korraldab loteriid, siis leiame kohe, et asi on kahtlane.

Sit näeme, et lisaks tõepärale on selleks, et me usuksime mõne teoria kehtimisse, vaja veel, et see teoria oleks piisavalt tõenäoline meie taustateadmiste valguses. Bayesi teoreem ei tee muud, kui arvutab teoria kehtimise posterioorse tõenäosuse (järeletõenäosuse), kasutades selleks meie eelteadmiste ja tõepära kvantitatiivseid mudeliteid. Seega, Bayesi paradigmas ei arvesta me mitte ainult andmetega, vaid ka taustateadmistega, sünteesides need kokku üheks posterioorseks jaotuseks ehk järeljaotuseks. Selle jaotuse arvutamine erineb bootstrapist, kuid tema tõlgendus ja praktiline töö sellega on sarnane. Erinevalt tavapärasest bootstrapist on Bayes parameetriline meetod, mis sõltub andmete modeleerimisest modeleerija poolt ette antud jaotustesse (normaaljaotus, t jaotus jne). Tegelikult peame me Bayesi arvutuseks modeleerima vähemalt kaks erinevat jaotust: andmete jaotus, mida me kutsume likelihoodiks ehk tõepäraks, ning eelneva teadmise mudel ehk prior, mida samuti modeleeritakse tõenäosusjaotusena.

Bootstrapil on mõned imelikud formaalsed eeldused: 1. väärtused, mis ei esine valimiandmetes, on võimatud, 2. Väärtused, mis esinevad väljaspool valimi väärustute vahemikku, on võimatud, 3. andmetes ei esine ajasõltuvusi ega hierarhilisi struktuure. Nendest puudustest hoolimata kasutatakse bootstrappimist laialt ja edukalt — eelkõige tema lihtsuse ja paindlikuse tõttu. Küll aga tähindab see, et bootstrap on harva parim võimalik meetod mingi ülesande lahendamiseks.

Ehkki bootstrappimine ei arvesta taustateadmistega, ei tee seda olulisel määral ka paljud Bayesi mudelid (mudeldaja vaba valiku tõttu, mitte selle pärast, et mudel ei suudaks taustainfot inkorporeerida). Bayesi meetodite väljatöötajad ei tea sageli ette, milliste teaduslike probleemide lahendamiseks nende mudeliteid hakatakse kasutama, ja seega ei kirjuta nad mudelisse ka väga ranget eelteadmist. Nende mudelite teadlastest kasutajad lepivad sageli selllega ja lasevad oma mudelite kaudu “andmetel kõneleda” enam-vähem sellistena, nagu need juhtuvad olema. Sellist lähenemist ei saa alati hukka mõista, sest vahest ei olegi meil palju eelteadmisi oma probleemi kohta, küll aga tuleb mainida, et sellistel juhtudel annab bootstrappimine sageli lihtsama vaevaga väga sarnase tulemuse kui Bayesi täismäng.

Chapter 13

Bayesi põhimõte

Bayesi arvutuseks on meil vaja teada

- 1) milline on “*parameter space*” ehk parameetriruum? Parameetriruum koosneb kõikidest loogiliselt võimalikest parameetriväärtustest. Näiteks kui me viskame ühe korra münti, koosneb parameetriruum kahest elemendist: 0 ja 1, ehk null kull ja üks kull. See ammendab võimalike sündmuste nimekirja. Kui me aga hindame mõnd pidevat suurust (keskmise pikkus, tõenäosus 0 ja 1 vahel jms), koosneb parameetriruum lõpmata paljudest elementidest (arvudest).
- 2) milline on “*likelihood function*” ehk tõepärafunktsioon? Me omistame igale parameetriruumi elemendile (igale võimalikule parameetri väärtsusele) tõepära. Tõepära parameetri väärtsel x on tõenäosus, millega me võiksime kohata oma andmete keskväärtust, juhul kui x oleks see ainus päris õige parameetri väärtsus. Teisisõnu, tõepära on kooskõla määr andmete ja parameetri väärtsuse x vahel. $\text{Tõepära} = P(\text{andmed} \mid \text{parameetri väärtsus})$. Näiteks, kui tõenäolised on meie andmed, kui USA keskmine president on juhtumisi 183.83629 cm pikkune? Kuna meil on vaja modeleerida tõepära igal võimalikul parameetri väärtsel (mida pideva suuruse puhul on lõpmatu hulk), siis kujutame tõepära pideva funktsionina (näiteks normaaljaotusena), mis täielikult katab parameetriruumi. Tõepärafunktsioon ei summeeru 100-le protsendile — see on normaliseerimata.
- 3) milline on “*prior function*” ehk prior? Igale tõepära väärtsusele peab vastama priori väärtsus. Seega, kui tõepära on modelleeritud pideva funktsionina, siis on ka prior pidev funktsioon (aga prior ei pea olema sama tüüpi funktsioon, kui tõepära). Erinevus tõepära ja priori vahel seisneb selles, et kui tõepärafunktsioon annab just meie andmete keskväärtuse tõenäosuse igal parameetriväärtusel, siis prior annab iga parameetriväärtuse tõenäosuse, sõltumata meie andmetest. See-eest arvestab prior kõikide teiste relevantsete andmetega, sünteesides taustateadmised ühte tõenäous-

mudelisse. Me omistame igale parameetriruumi väärtsusele eelneva tõenäosuse, et see väärtsus on üks ja ainus tõene väärtsus. Prior jaotus summeerub 1-le. Prior kajastab meie konkreetsetest andmetest sõltumatut arvamust, kui suure tõenäosusega on just see parameetri väärtsus tõene; seega seda, mida me usume enne oma andmete nägemist. Nendel parameetri väärtsustel, kus prior ($v\ddot{o}$ i tõepära) = 0%, on ka posteerior garantteeritult 0%. See tähendab, et kui te olete 100% kindel, et mingi sündmus on võimatu, siis ei suuda ka mäekõrgune hunnik uusi andmeid teie uskumust muuta (eelduselt, et te olete ratsionaalne inimene).

<http://optics.eee.nottingham.ac.uk/match/uncertainty.php> aitab praktikas priorit modelleerida (proovige *Roulette* meetodit).

Kui te eelnevast päriselt aru ei saanud, ärge muretsege. Varsti tulevad puust ja punaseks näited likelihood ja priori kohta.

Edasi on lihtne. Arvuti võtab tõepärafunktsiooni ja priori, korrutab need üksteisega läbi ning seejärel normaliseerib saadud jaotuse nii, et jaotusealune pindala võrdub ühega. Saadud tõenäosusjaotus ongi posteeriorne jaotus ehk posteerior ehk järeljaotus. Kogu lugu.

Me teame juba pool sajandit, et Bayesi teoreem on sellisele ülesandele parim võimalik lahendus. Lihtsamad ülesanded lahendame me selle abil täiuslikult. Kuna parameetrite arvu kasvuga mudelis muutub Bayesi teoreemi läbiarvutamine eksponentsialselt arvutusmahukamaks (sest läbi tuleb arvutada mudeli kõikide parameetrite kõikide väärtsuste kõikvõimalikud kombinatsioonid), oleme sunnitud vähegi keerulisemad ülesanded lahendama umbkaudu, asendades Bayesi teoreemi *ad hoc* MCMC algoritmiga, mis teie arvutis peituva propelleri Karlsoni kombel lendu saadab, et tõmmata valim “otse” posterioorset jaotusest. Meie poolt kasutatava MCMC *Hamiltonian Monte Carlo* mootori nimi on Stan (www.mc-stan.org). See on eraldiseisev programm, millel on R-i liides R-i pakettide rstan(), rethinking(), rstanarm() jt kaudu. Meie töötame ka edaspidi puhtalt R-s, mis automaatselt suunab meie mudelid ja muud andmed Stani, kus need läbi arvutatakse ja seejärel tulemused R-i tagasi saadetakse. Tulemuste töötlus ja graafiline esitus toimub jällegi R-is. Seega ei pea me ise kordagi Stani avama.

Alustame siiski lihtsa näitega, mida saab käsitsi läbi arvutada.

Esimene näide

Me teame, et suremus haigusesse on 50% ja meil on palatis 3 patsienti, kes seda haigust põevad. Seega on meil kaks andmetükki (50% ja n=3). Küsimus: mitu meie patsienti oodatavalt hinge heidavad? Eeldusel, et meie patsiendid on iseseisvad (näiteks ei ole sugulased), on meil tüüpiline mündiviske olukord.

Parameetriruum on neljaliikmeline: 0 surnud, 1 surnud, 2 surnud ja 3 surnud. Edasi loeme üles kõik võimalikud sündmusteahelad, mis loogiliselt saavad juhtuda,

et saada tõepärafunktsioon.

Me viskame kulli-kirja 3 korda: kiri = elus, kull = surnud

Võimalikud sündmused on: | kull kull kull | kull kiri kull | kiri kull kull | kull
kull kiri | kull kiri kiri | kiri kiri kull | kiri kull kiri | kiri kiri kiri

Kui $P(\text{kull}) = 0.5$, siis lugedes kokku kõik võimalikud sündmused:

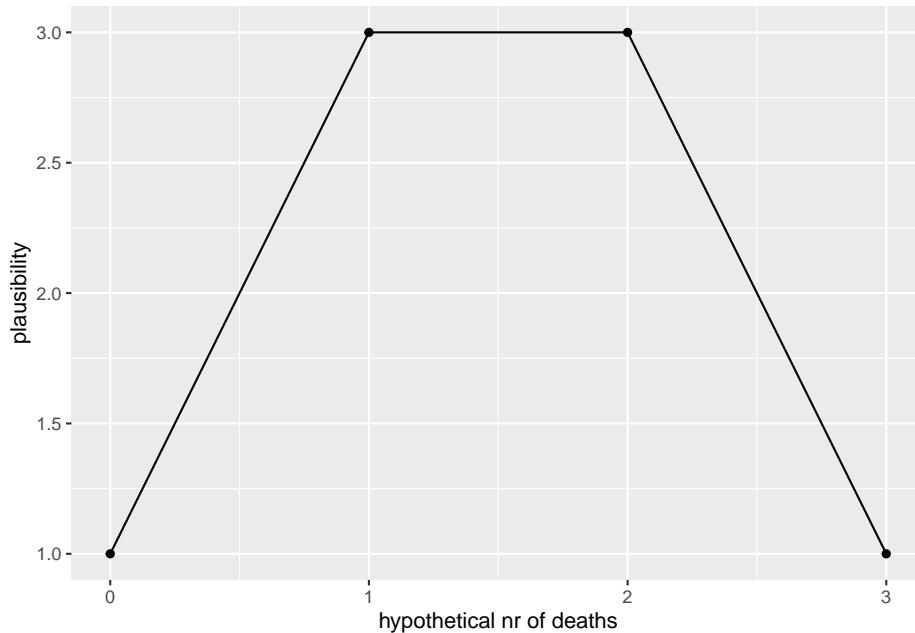
- 0 kulli ehk surnud - 1,
- 1 kulli ehk surnud - 3,
- 2 kulli ehk surnud - 3,
- 3 kulli ehk surnud - 1

Nüüd teame parameetrituumi iga liikme kohta, kui suure tõenäosusega me ootame selle realiseerumist. Näiteks, $P(0 \text{ surnud}) = 1/8$, $P(1 \text{ surnud}) = 3/8$, $P(1 \text{ või } 2 \text{ surnud}) = 6/8$ jne Selle teadmise konverteerime tõepärafunktsioniks.

```
library(tidyverse)
# Parameter space: all possible futures
x <- seq(from = 0, to = 3)

# Likelihoods for each x value, or P(deaths | x)
y <- c(1, 3, 3, 1)

ggplot(data=NULL, aes(x, y)) +
  geom_point()+
  geom_line()+
  xlab("hypothetical nr of deaths")+
  ylab("plausibility")
```

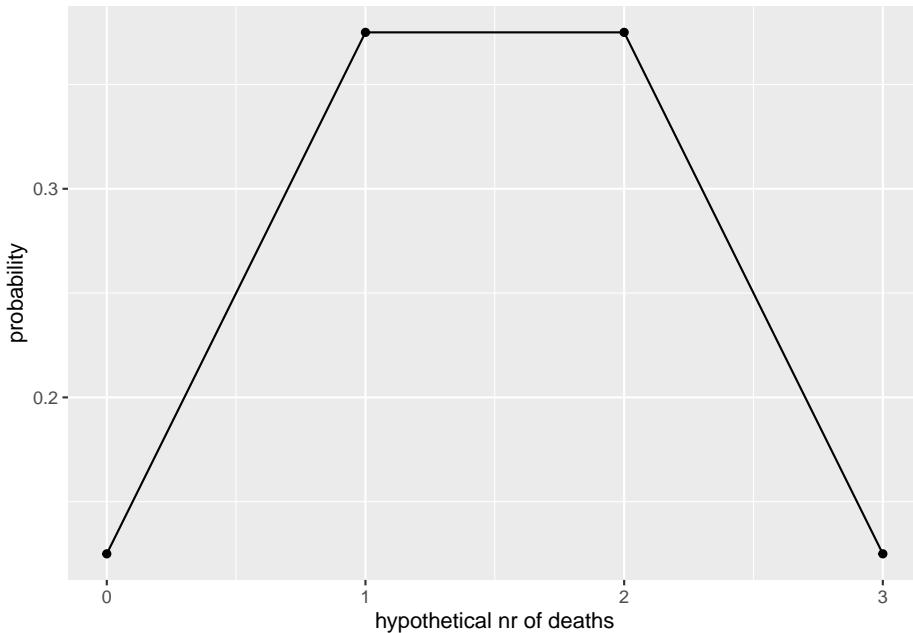


Sit näeme, et üks surm ja kaks surma on sama tõenäolised ja üks surm on kolm korda tõenäolisem kui null surma (või kolm surma). Tõepära annab meile tõenäosuse $\Pr(\text{mortality}=0.5 \& N=3)$ igale loogiliselt võimalikule surmade arvule (0 kuni 3).

Me saame sama tulemuse kasutades formaalsel viisil binoomjaotuse mudelit. Ainus erinevus on, et nüüd on meil y teljel surmade tõenäosus.

```
y <- dbinom(x, 3, 0.5)

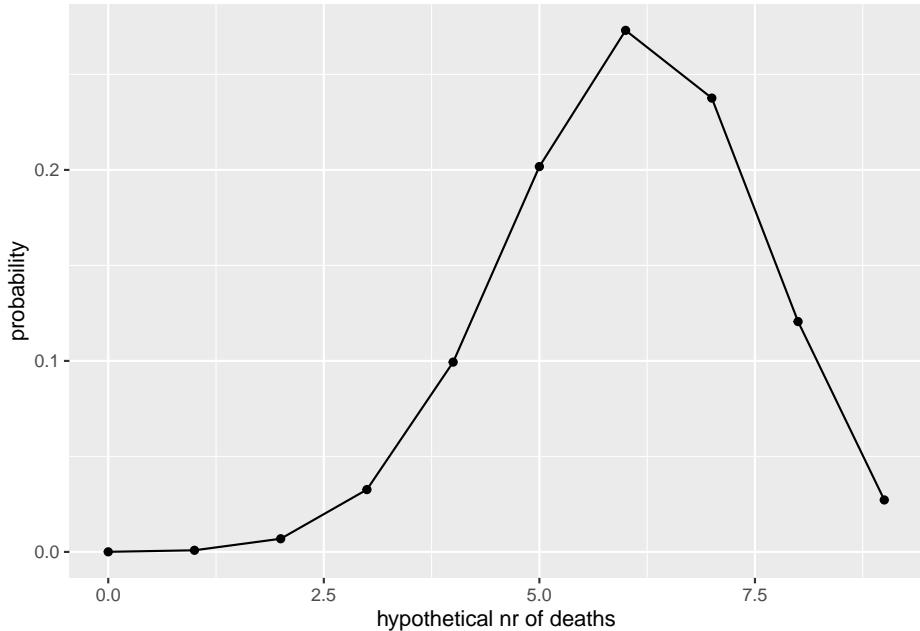
ggplot(data=NULL, aes(x, y)) +
  geom_point() +
  geom_line() +
  xlab("hypothetical nr of deaths") +
  ylab("probability")
```



Proovime seda koodi olukorras, kus meil on 9 patsienti ja suremus on 0.67:

```
x <- seq(from = 0, to = 9)
y <- dbinom(x, 9, 0.67)

ggplot(data=NULL, aes(x, y)) +
  geom_point()+
  geom_line()+
  xlab("hypothetical nr of deaths")+
  ylab("probability")
```



Lisame sellele tõepärafunktsioonile tasase priori (lihtsuse huvides) ja arvutame posterioorse jaotuse kasutades Bayesi teoreemi. Igale parameetri väärustusele on tõepära * prior proporsionaalne posterioorse töenäosusega, et just see parameetri väärustus on see ainus töene väärustus. Posterioorsed töenäosused normaliseeritakse nii, et nad summeeruksid 1-le.

Me defineerime X telje kui rea 10-st arvust (0 kuni 9 surma) ja arvutame tõepära igale neist 10-st arvust. Sellega ammendame me kõik loogiliselt võimalikud parameetri väärusted.

```
x <- seq(from = 0 , to = 9)
# flat prior
prior <- rep(1 , 10)

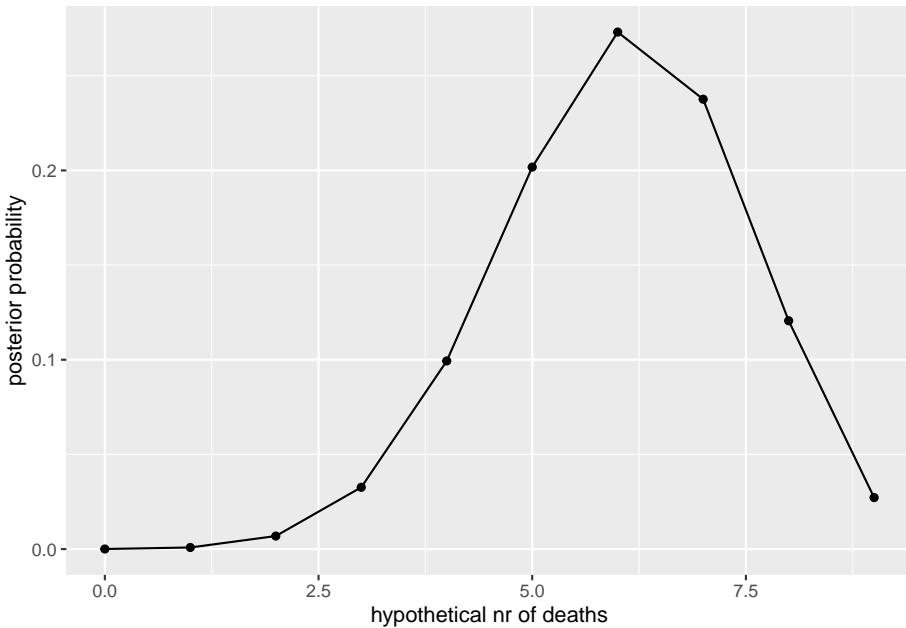
# Compute likelihood at each value in grid
likelihood <- dbinom(x, size = 9, prob = 0.67)

# Compute product of likelihood and prior
unstd.posterior <- likelihood * prior

# Normalize the posterior, so that it sums to 1
posterior <- unstd.posterior / sum(unstd.posterior)

ggplot(data = NULL, aes(x, posterior)) +
  geom_point() +
  geom_line()
```

```
xlab("hypothetical nr of deaths") +
ylab("posterior probability")
```



See on parim võimalik teadmine, mitu kirstu tasuks tellida, arvestades meie priori ja likelihoodi mudelitega. Näiteks, sedapalju, kui surmad ei ole üksteisest sõltumatud, on meie tõepäramudel (binoomjaotus) vale.

Teine näide: sõnastame oma probleemi ümber

Mis siis, kui me ei tea suremust ja tahaksime seda välja arvutada? Kõik, mida me teame on, et 6 patsienti 9st surid. Nütüd koosnevad andmed 9 patsiendi mortaalsusinfost (parameeter, mille väärust me eelmises näites arvutasime) ja parameeter, mille väärust me ei tea, on surmade üldine sagedus (see parameeter oli eelmises näites fikseeritud, ja seega kuulus andmete hulka).

Seega on meil:

1. Parameetriruum 0% kuni 100% suremus (0st 1-ni), mis sisaldab lõpmata palju numbreid.
2. Kaks võimalikku sündmust (surnud, elus), seega binoomjaotusega modelleeritud tõepärafunktsioon. Nagu me juba teame, on r funktsionis dbinom() kolm argumenti: surmade arv, patsientide koguarv ja surmade tõenäosus. Seekord oleme me fikseerinud esimesed kaks ja soovime arvutada kolmanda väärtsuse.

3. Tasane prior, mis ulatub 0 ja 1 vahel. Me valisime selle priori selleks, et mitte muuta tõepärafunktsiooni kuju. See ei tähenda, et me arvaksime, et tasane prior on mitteinformatiivne. Tasane prior tähendab, et me usume, et suremuse kõik vääratused 0 ja 1 vahel on võrdselt tõenäolised. See on vägagi informatsioonirohke (ebatalvine) viis maailma näha, ükskõik mis haiguse puhul!

Tõepära parameetri vääratusel x on tõenäosus kohata meie andmeid, kui x on juhtumisi parameetri tegelik vääratus. Meie näites koosneb tõepärafunktsioon tõenäosustest, et kuus üheksast patsiendist surid igal võimalikul suremuse vääratusel ($0 \dots 1$). Kuna see on lõpmatu rida, teeme natuke sohki ja arvutame tõepära 20-l valitud suremuse vääratusel.

Tehniliselt on sinu andmete tõepärafunktsioon agregeeritud iga üksiku andmepunkti tõepärafunktsionist. Seega vaatab Bayes igat andmepunkti eraldi (andmete sisestamise järvjekord ei loe).

```
# mortality at 20 evenly spaced probabilities from 0 to 1
x <- seq(from = 0 , to = 1, length.out = 20)

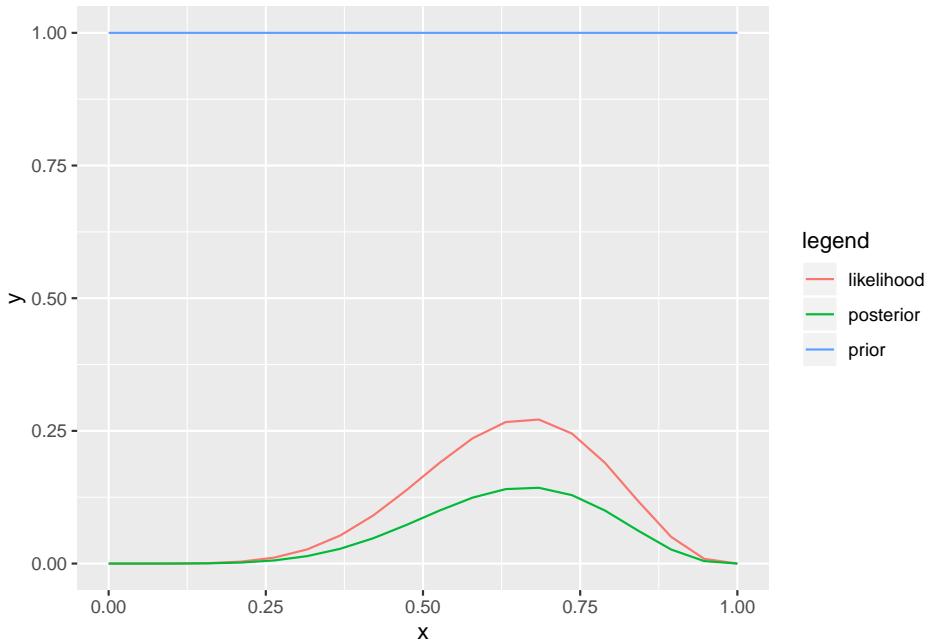
# Define prior
prior <- rep(1 , 20)

# Compute likelihood at each value in grid
likelihood <- dbinom(6, size = 9 , prob = x)

# Compute product of likelihood and prior & standardize the posterior
posterior <- likelihood * prior / sum(likelihood * prior)

#put everithing into a tibble for plotting
a <- tibble(x=rep(x= x, 3),
            y= c(prior, likelihood, posterior),
            legend= rep(c("prior","likelihood", "posterior"), each=20))

ggplot(data=a) + geom_line(aes(x, y, color=legend))
```



Nüüd on meil posterioorne tõenäosusfunktsioon, mis summeerub 1-le ja mis sisaldab kogu meie teadmist suremuse kohta.

Alati on kasulik plottida kõik kolm funktsiooni (tõepära, prior ja posteerior).

Kui $n = 1$

Bayes on lahe sest tema hinnangud väiksele N -le on loogiliselt sama pädevad kui suurele N -le. See ei ole nii klassikalises sageduslikus statistikas, kus paljud testid on välja töötatud $N = \text{Inf}$ eeldusel ja töötavad halvasti väikeste valimitega.

Hea küll, me arvutame jälle suremust.

Bayes töötab andmepunkti kaupa (see et me talle ennist kõik andmed korraga ette andsime, on puhtalt mugavuse pärast).

```
# Define grid
x <- seq(from = 0, to = 1, length.out = 20)

# Define prior
prior <- rep(1, 20)

# Compute likelihood at each value in grid
likelihood <- dbinom(1, size = 1, prob = x)
```

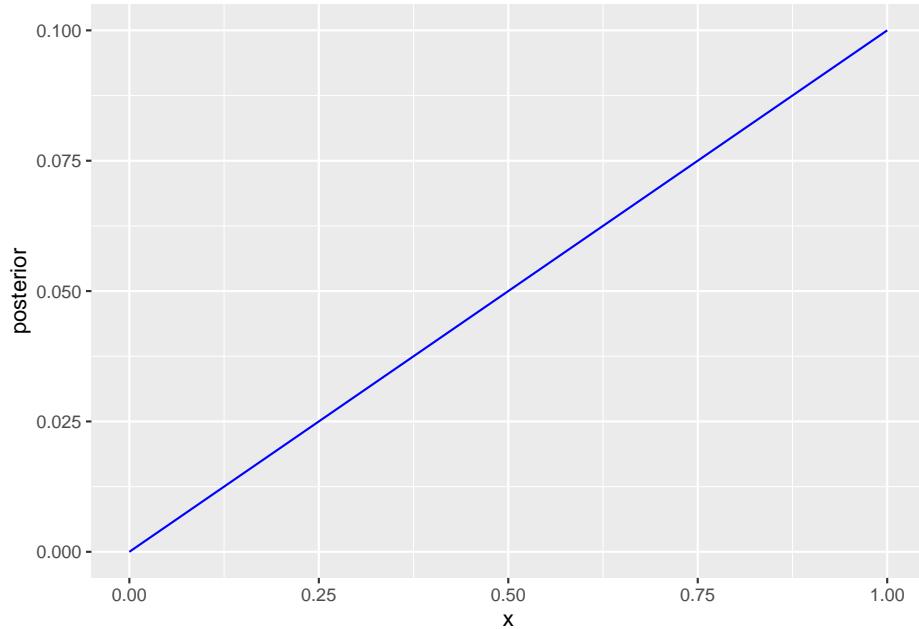


Figure 13.1: N=1, esimene patsient suri.

```
posterior <- likelihood * prior / sum(likelihood * prior)

ggplot(data=NULL) +
  geom_line(aes(x, posterior), color= "blue")
```

Esimene patsient suri - 0 mortaalsus ei ole enam loogiliselt võimalik (välja arvatud siis kui prior selle koha peal = 0) ja mortaalsus 100% on andmetega (tegelikult andmega) parimini kooskõlas. Posteeriор on null ja 100% vahel sirge sest vähene sissepandud informatsioon lihtsalt ei võimalda enamat.

```
x <- seq(from = 0, to = 1, length.out = 20)
# Define prior
prior <- posterior

# Compute likelihood at each value in grid
likelihood <- dbinom(1 , size = 1, prob = x)
posterior1 <- likelihood * prior / sum(likelihood * prior)

ggplot(data=NULL) +
  geom_line(aes(x, prior)) +
  geom_line(aes(x, posterior1), color="blue")
```

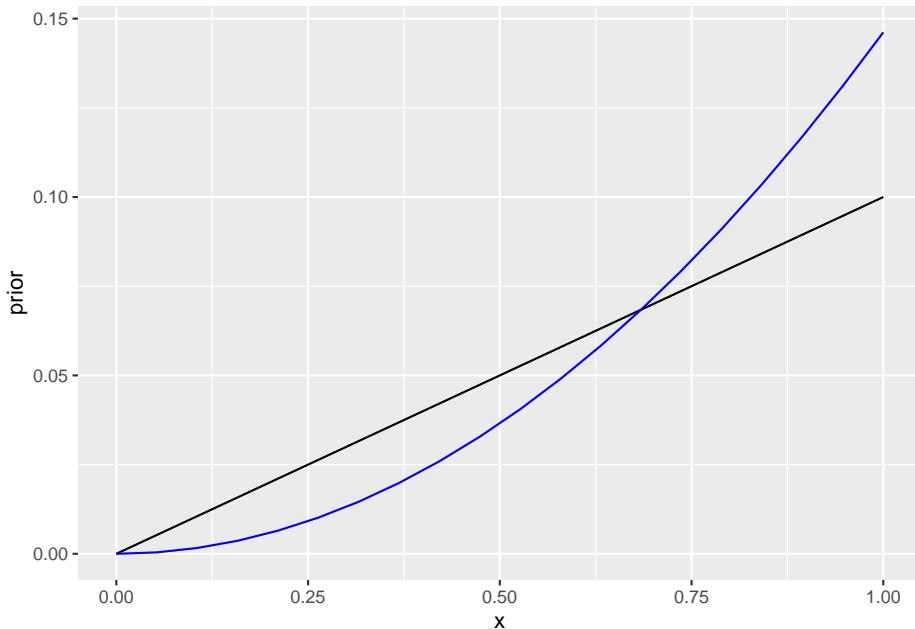


Figure 13.2: N=2, teine patsient suri.

Teine patsient suri. Nüüd ei ole 0 ja 1 vahel enam sirge posteerior. Posteerior on kaldo 100 protsendi poole, mis on ikka kõige tõenäolisem vääratus.

```
x <- seq(from = 0, to = 1, length.out = 20)
# Define prior
prior <- posterior1

# Compute likelihood at each value in grid
likelihood <- dbinom(0, size = 1, prob = x)

# Compute product of likelihood and prior
posterior2 <- likelihood * prior / sum(likelihood * prior)

ggplot(data=NULL) +
  geom_line(aes(x, prior)) +
  geom_line(aes(x, posterior2), color="blue")
```

Kolmas patsient jäi ellu - 0 ja 100% mortaalsus on seega võimaluste nimekirjast maas ning suremus on ikka kaldo valimi keskmise poole (75%).

Teeme sedasama prioriga, mis ei ole tasane. See illustreerib tõsiasja, et kui N on väike siis domineerib prior posteerior jaotust. (Suure N korral on vastupidi,

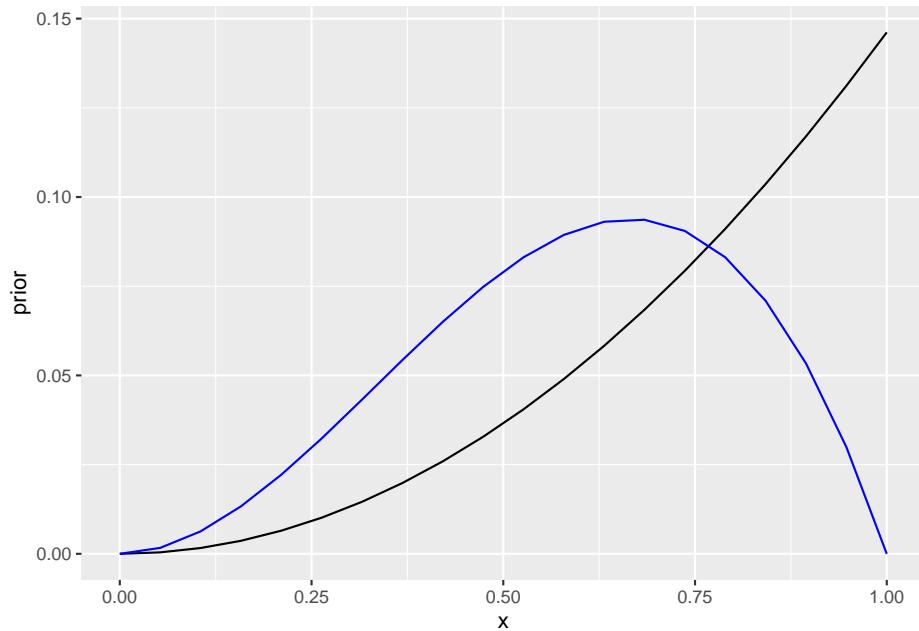


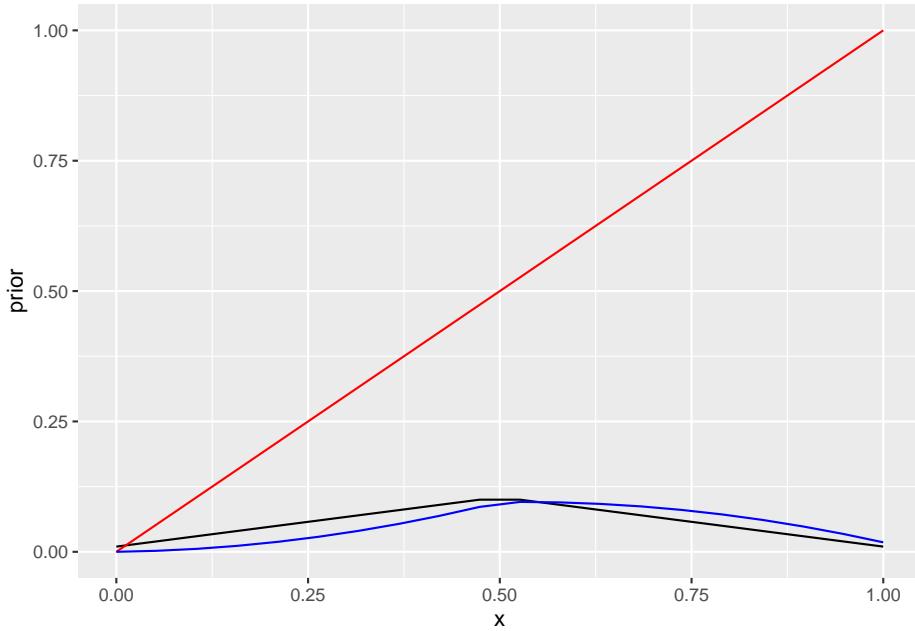
Figure 13.3: N=3, kolmas patsient jäi ellu.

priori kuju on sageli vähetähtis.)

```
x <- seq(from = 0, to = 1, length.out = 20)
# Define prior
prior <- c(seq(0.01, 0.1, length.out = 10), seq(0.1, 0.01, length.out = 10))

# Compute likelihood at each value in grid
likelihood <- dbinom(1, size = 1, prob = x)
posterior <- likelihood * prior / sum(likelihood * prior)

ggplot(data=NULL) +
  geom_line(aes(x, prior)) +
  geom_line(aes(x, likelihood), color="red") +
  geom_line(aes(x, posterior), color="blue")
```



1. patsient suri

```
# Define prior
prior <- posterior

# Compute likelihood at each value in grid
likelihood <- dbinom(2, size = 2, prob = x)

# Compute product of likelihood and prior
posterior1 <- likelihood * prior / sum(likelihood * prior)

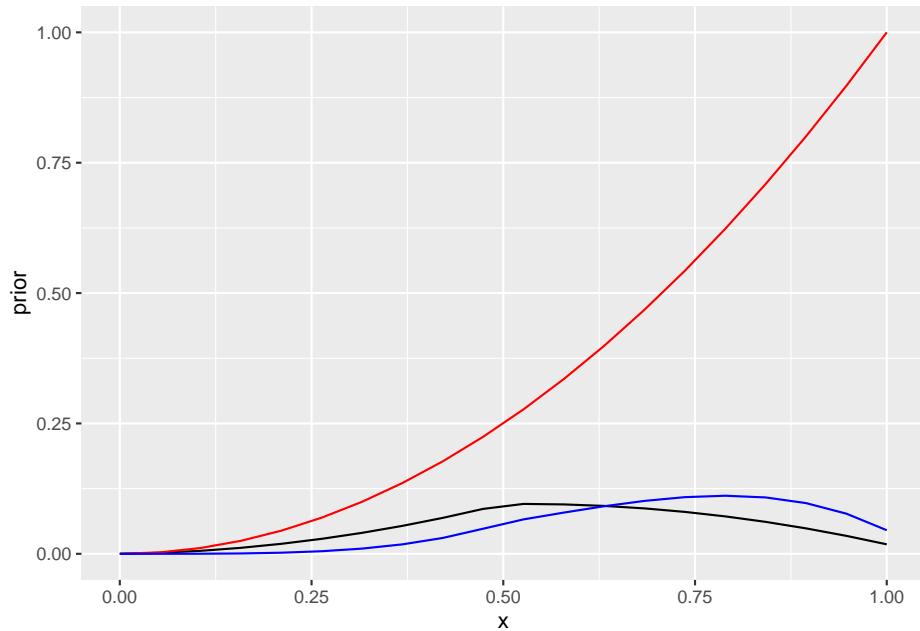
ggplot(data=NULL)+
  geom_line(aes(x, prior))+
  geom_line(aes(x, likelihood), color="red")+
  geom_line(aes(x, posterior1), color="blue")
```

Teine patsient suri.

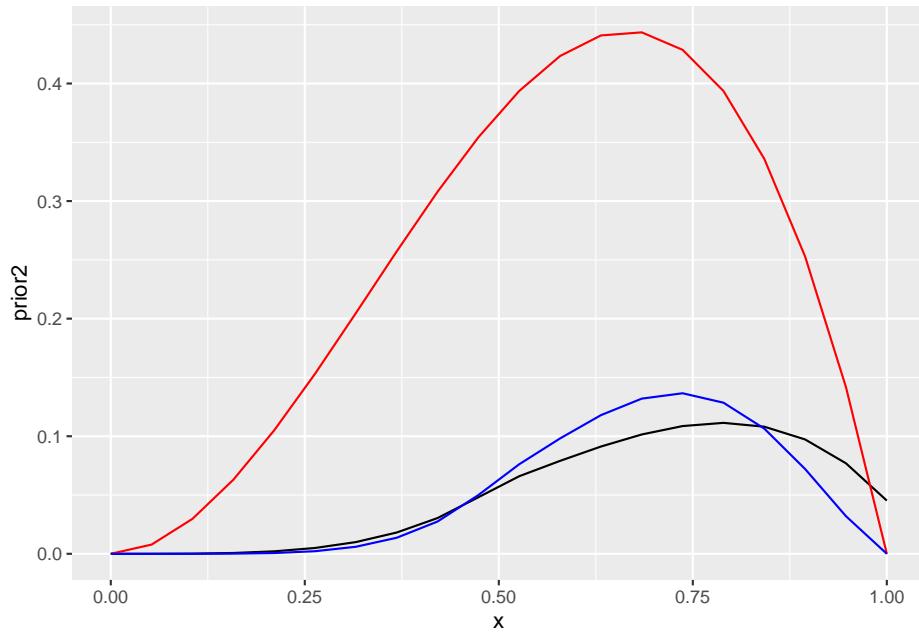
```
# Define prior
prior2 <- posterior1

# Compute likelihood at each value in grid
likelihood <- dbinom(2, size = 3, prob = x)

# Compute product of likelihood and prior
posterior2 <- likelihood * prior2 / sum(likelihood * prior2)
```

Figure 13.4: $N=2$ informatiivse prioriga.

```
ggplot(data=NULL)+  
  geom_line(aes(x, prior2))+  
  geom_line(aes(x, likelihood), color="red") +  
  geom_line(aes(x, posterior2), color="blue")
```



Kolmas patsient jäi ellu. Nüüd on posteeriori tipp mitte 75% juures nagu ennist, vaid kuskil 50% juures — tänu priorile.

Chapter 14

Mudelite keel

Siin vaatame kuidas kirjeldada mudelit nii, et masin selle ära tunneb. Meie mudelid töötavad läbi “rethinking” paketi. See raamatukogu pakub kaks võimalust, kuidas mudelit arvutada, mis mõlemad kasutavad sama notatsiooni. Mõlemad võimalused arvutavad posteeriori mitte Bayesi teoreemi kasutades (nagu me ennist tegime), vaid kasutades stohastilisi meetodeid, mis iseloomustavad posteeriori umbkaudu (aga piisavalt täpselt). Põhjuseks on, et keerulisemate mudelite korral on Bayesi teoreemi kasutamine liialt arvutusmahukas.

Esiteks `rethinking::map()` leiab posteeriori tipu ja selle lähedal funktsiooni tõusunurga. Siin on eelduseks, et posteerior on normaaljaotus. See eeldus kehtib alati, kui nii prior kui tõepära on modelleeritud normaaljaotusena (ja ka paljudel muudel juhtudel).

Teine võimalus on `rethinking::map2stan()`, mis suunab teie kirjutatud mudeli Stan-i. Stan teeb *Hamiltonian Monte Carlo* simulatsiooni, kasutades valget maagiat selleks, et tõmmata valim otse posteeroorsest jaotusest. See on väga moodne lähenemine statistikale, töötab oluliselt aeglasemalt kui `map`, aga ei sõltu normaaljaotustest ning suudab arvutada hierarhilisi mudeliteid, mis `map`-le üle jõu käivad.

Me võime sama mudeli kirjelduse sööta mõlemasse funktsiooni.

Lihtne mudel näeb välja niimodi:

```
dead ~ dbinom(9, p) # binomial likelihood  
p ~ dunif(0, 1) # uniform prior
```

Tõepärafunktsioon on modeleeritud binoomjaotusena. Parameeter, mille väärust määräatakse on `p`, ehk suremus. See on ainus parameeter, mille väärust me siin krutime. NB! igale parameetrile peab vastama oma prior. Meil on selles mudelis täpselt 1 parameeter ja 1 prior. Vastuseks saame selle ainsa parameetri posterioorse jaotuse. Hiljem näeme, et kui meil on näiteks 452 parameetrit,

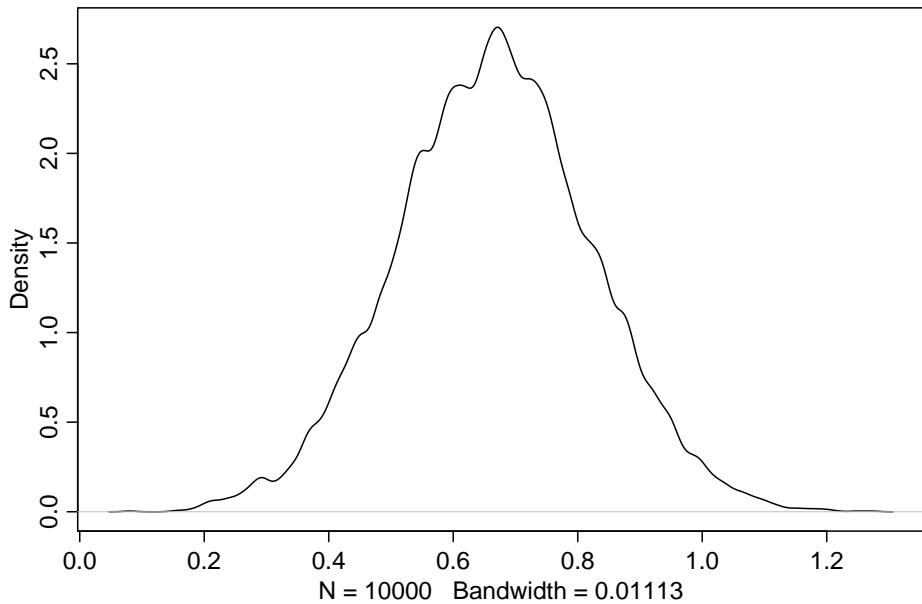


Figure 14.1: Tõmbame valimi posteeriorist.

mille väärtsusi me koos arvutame, siis on meil ka 452 priorit ja 452 posteerioorset jaotust.

```
library(rethinking)
# Fit model using rethinking
m1 <- map(
  alist(
    dead ~ dbinom(9, p), # Binomial likelihood
    p ~ dunif(0, 1) # Uniform prior
  ), data = list(dead = 6))

# Summary of quadratic approximation
precis(m1)
#>   Mean StdDev 5.5% 94.5%
#>   p  0.67   0.16  0.42  0.92
```

Nüüd tõmbame posteeriooroorest jaotusest valimi $n=10\ 000$. Selleks on funktsioon `extract.samples()`

```
samples <- extract.samples(m1)
# hist(samples$p)
dens(samples$p)
```

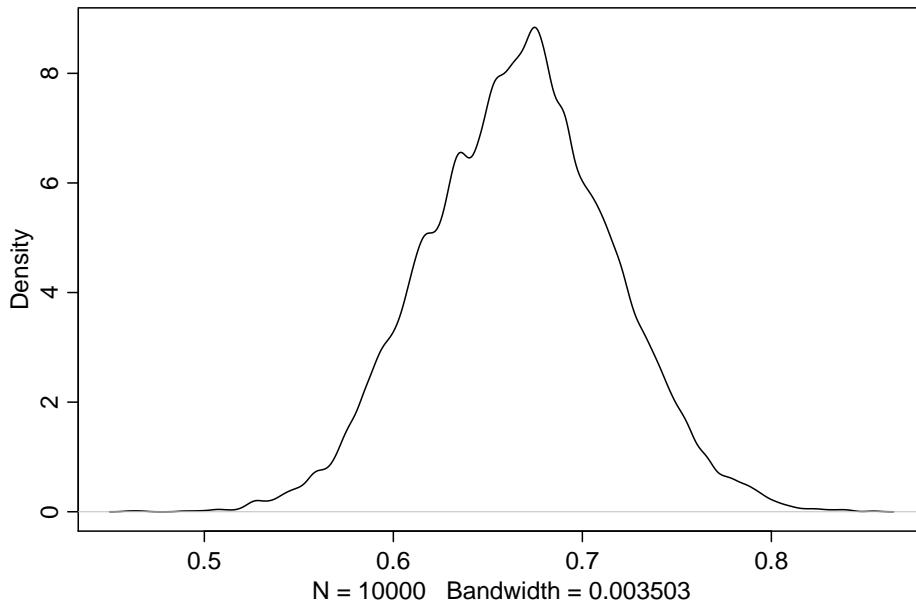


Figure 14.2: Veel üks valim posteeriorist (60 surma 90st).

```
HPDI(samples$p, prob = 0.95) # Highest density 95% at the center
#> 10.95 0.95/
#> 0.357 0.972
```

Kuus patsienti üheksast surid ja nüüd me usume, et tegelik suremus võib olla nii madal kui 37% ja nii kõrge kui 97%. Kui me tahame paremat hinnangut on meil vaja kas rohkem patsiente või informatiivsemat priorit (paremat taustainfot).

```
m2 <- map(
  alist(
    dead ~ dbinom(90, p), # Binomial likelihood
    p ~ dunif(0, 1)      # Uniform prior
  ), data = list(dead = 60))
# Display summary of quadratic approximation
precis(m2)
#> Mean StdDev 5.5% 94.5%
#> p 0.67 0.05 0.59 0.75

samples <- extract.samples(m2)
dens(samples$p)

#PI(samples$p, prob = 0.95) # Leaves out equal 2.5% at both sides
HPDI(samples$p, prob = 0.95) # Highest density 95% at the center
```

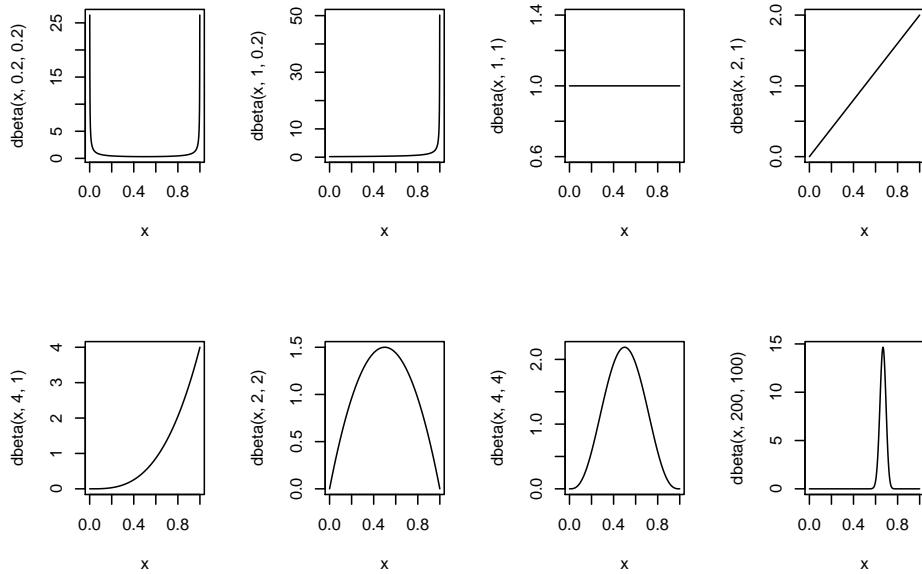


Figure 14.3: Beta jaotuse parametriseeringuid.

```
#> /0.95 0.95/
#> 0.57 0.76
```

10 korda rohkem andmeid: nüüd on suremus määratud kuskile 57% ja 77% vaheline (suure tõenäosusega)

Beta prior

Nüüd anname sisse mõistlikuma struktuuriga priori: beta-jaotuse

Beta-prior katab vahemiku 0st 1ni ja sellel on 2 parameetrit, a ja b .

Siin mõned näited erinevatest beta parametriseeringutest.

$\text{beta}(\theta | a, b)$ jaotuse keskväärtus on

$$\mu = a/(a + b)$$

ja mood on

$$\omega = (a - 1)/(a + b - 2) \text{ (kui } a > 1 \text{ ja } b > 1\text{).}$$

Seega, kui $a = b$, siis on keskmise ja mood 0.5. Kui $a > b$, on keskmise ja mood > 0.5 ja kuid $a < b$, on mõlemad < 0.5 .

Beta jaotuse “laiuse” annab “kontsentratsioon” $\kappa = a + b$. Mida suurem κ , seda kitsam jaotus.

$$a = \mu\kappa$$

$$b = (1 - \mu)\kappa$$

$$a = \omega(\kappa - 2) + 1$$

$$b = (1 - \omega)(\kappa - 2) + 1 \text{ kui } \kappa > 2$$

Me võime κ -le omistada väärtsuse nagu see oleks mündivisete arv, mis iseloomustab meie priori tugevust (juhul kui tõepära funktsioon tuleb andmetest, mis koosnevad selle sama mündi visetest). Kui meie jaoks piisaks ainult mõnest mündivisest, et priorit (eelnevast teadmisest) lahti ütelda, peaks meie prior sisaldama väikest kappat.

Näiteks, mu prior on, et münt on aus ($\mu = 0.5$; $a = b$), aga ma ei ole selles väga veendunud. Niisiis ma arvan, et selle eelteadmise kaal võrdub sellega, kui ma oleksin näinud 8 mündiviske tulemust. Seega $\kappa = 8$, mis tähendab, et $a = \mu\kappa = 4$ ja $b = (1 - \mu)\kappa = 4$. Aga mis siis kui me tahame beta priorit, mille mood $\omega = 0.8$ ja $\kappa = 12$? Siis saame valemist, et $a = 9$ ja $b = 3$.

```
## Fit model
m3 <- rthinking::map(
  alist(
    dead ~ dbinom(9, p), # Binomial likelihood
    p ~ dbeta(200, 100) # Beta prior
  ), data = list(dead = 6))
## Extract samples
samples <- extract.samples(m3)
# Display summary of quadratic approximation
precis(m3)
#>   Mean StdDev 5.5% 94.5%
#> p 0.67  0.03 0.62  0.71

HPDI(samples$p, prob = 0.95) # Highest density 95% at the center
#> /0.95 0.95/
#> 0.616 0.720
dens(samples$p)
```

Nagu näha on ka kitsa priori mõju üsna väika, isegi kui kui $n = 9$.

Prioritest üldiselt

Neid võib jagada kolmeksi: mitteinformatiivsed, väheinformatiivsed ehk “regularizing” ja informatiivsed.

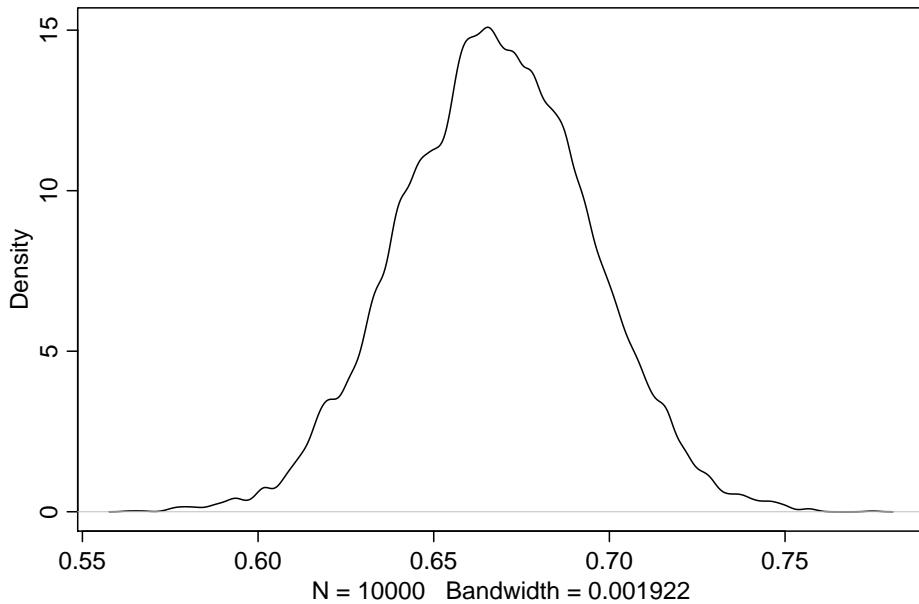


Figure 14.4: Posteeriор, mis on arvutatud beta prioriga binoomsest tõepära-mudelist.

Mitteinformatiivseid prioreid ei ole sisuliselt olemas ja neid on soovitav välida. Sageli kutsutakse tasaseid prioreid mitteinformatiivseteks. Neil on vähemalt 2 puudust. Tasane prior, mis ulatub lõpmatusse, on tehniliselt “improper”, sest tema alune pindala ei summeeru ühele. Ja teiseks muudavad sellised priorid mcmc ahelad vähem efektiivseteks, mis võib teie arvutuse kihva keerata.

Väheinformatiivsed priorid kujutavad endast kompromissi: nad muudavad võimalikult vähe tõepärafunktsiooni kuju, aga samas piiravad seda osa parameetrituruumist, kust MCMC ahelad posteeriорi otsivad (mis on soodne arvutuslikult). Nende priorite taga on filosoofiline eeldus, et teadlast huvitavad eelkõige tema enda andmed ja see, mida need ühe või teise hüpoteesi (parameetri väärtsuse) kohta ütlevad. See eeldus on vaieldav, aga kui selle järgi käia, siis kulub vähem mõttejõudu eelteadmiste mudelisse formaliseerimiseks.

Nõrgalt regulariseerivad priorid on väheinformatiivsete priorite alamliik, mis on tsentreeritud nullile ja tömbavad posteeriорit õrnalt null suunas. Regulariseerimine tähendab seda, et kui regressiooni beta-koefotsiendi prior surub seda koefitsienti õrnalt null poole (nulltõus tähendab efekti puudumist), siis reageerib mudel selle võrra vähem andmetele, mis kallutavad seda betat nullist eemale, ja tulemuseks on loodetavasti mudeli ülefittimise vältimine. Selline lähenemine töötab päriselt, aga samas pole alati lihtne öelda, kui palju me peaksime konkreetsel juhul prioritega regulariseerima - üleregulariseeritud mudel

ei arvesta piisavalt andmetega, mis on selgelt raiskamine.

Vähemalt suured farmaatsiafirmad seda hoiakut ei jaga ja kulutavad oma miljoneid informatiivsete priorite tootmiseks. Selles protsessis saavad kokku statistikud, teadusekspertid ja psühholoogid, et inimkonna teadmisi võimalikult adekvaatselt vormida tõenäosusjaotustesse. Meie töötame siiski enamasti väheinformatiivsete prioritega.

Chapter 15

MCMC põhimõte

MCMC tähendab Markovi ahelate Monte Carlo meetodit. Sellel meetodil on siis kaks poolt: Markovi ahelad ja Monte Carlo. **Monte Carlo** on simulatsioonimeetod, mis kasutab juhuslike arvude genereerimist parameetriväärtuse umbkaudseks hindamiseks. Näiteks, et hinnata ebaregulaarse kujundi pindala, (i) joonista selle ümber ruut, mille pindala sa oskad mõõta, (ii) viska sellele ühendkujundile juhuslikult n palli, millest osad langevad algsele kujundile aga kõik langevad ruutu; (iii) nende pallide proportsioon korrutatuna ruudu pindalaga annab teile algse kujundi pindala.

Markovi ahel kujutab endast üksteisega tõenäosuslikult seotud sündmuste järjestust. Markovi ahel liigub sammhaaval mõõda fikseeritud hulka võimalikke sündmusi (parameetriruumi), kusjuures iga juba toiminud samm määrab järgmiste võimalike sammude tõenäosused. Igal sammul peatub ahel ühel sündmusel (parameetriväärtusel) ja seejärel viib järgmine samm uuele parameetriväärtusele, või mitte (ahel jäääb paigale). See süsteem töötab ilma mäluta: järgmine samm sõltub sellest, millisel parameetriväärtusel on ahel praegu, mitte süsteemi ajaloost.

Lihtsaim ja ajalooliselt esimene Markovi ahela tüüp on Metropolise algoritm. Kujutage endale ette 1-mõõtmelist parameetriruumi, mis vastab ühe parameetriga mudelile, kus iga punkt kujutab endast ühte parameetriväärtust. Ahel maandub sellel 1-D sirgel juhuslikus punktis, misjärel on tal võrdne võimalus vaadata vasakule või paremale. Ta valib juhuslikult ühe parameetriväärtuse ehk prospekti, aga selle asemel, et sinna hüputa, hoopis kaalutleb meie andmetele ja priorile toetudes, milline on selle prospekti tõenäolisus vörreldes ahela praeguse positsiooniga. Ehk, kui palju kordi on prospekt andmete & priori poolt rohkem toetatud kui käesolev parameetriväärtus. Kui prospekt on näiteks 2 korda paremini toetatud, siis liigub ahel tõenäosusega $2/3$ uuele positsioonile ja tõenäosusega $1/3$ jäääb paigale.

Sellisel viisil edasi-tagasi sammudes veedab ahel rohkem aega parameetriruumi

piirkonnas, mis on andmetega paremas kooskõlas, ja pikas perspektiivis annavad ahela sammud juhuvalimi posteeriorist. Ahela esimesed tuhat sammu loetakse nn sissepõletamise perioodiks, mil ahel otsib katse-eksituse meetodil posteeriori tihedamat (tõenäolisemat) ala, ja neid ei salvestata posterioorse valimi hulka. Salvestatud ahela osa hõlmab tavaliselt mõned tuhanded sammud. Sageli jooksutame paraleelselt 3 või 4 iseseisvat ahelat ja vaatame, kas need konvergeeruvad samas parameetrväärtuste piirkonnas. See on tähtis kvaliteedinäitaja - kui mõni ahel liigub teistest eemal, siis ei saa me oma arvutust usaldada.

Seega võtab ahel posteeriorist juhuslikke arve, millega me saame hiljem otse töötada - näiteks plottida posteeriori histogrammi ja leida sealta suurima tiheusega piirkonna, kuhu jäab 95% ahela sammudest (ehk 95% CI ehk 95% HDI [highest density interval]). Kui meie mudelis on n parameetrit, siis jookseb ahel n -dimensionaalses matemaatilises ruumis, kus osad parameetrväärtused on andmetega paremas kooskõlas kui teised. Tänapäeval jooksutatakse ka paarikümne tuhande parameetriga mudeleid, mille jaoks arvutil kulub tavaliselt kuni paar päeva.

Metropolise algoritm leiab peale paljude sammude astumist garanteeritult õige posteeriori ja võtab sellest juhuvalimi. Probleem on siin selles, kui palju samme selleks tegelikkuses kulub. Kuna algoritmil on võimalus ka paigale jäädva, siis olukorras, kus posteerior asub ahela praegusest asukohast kaugel, või on väga kitsas (hõlmab vaid tühist osa parameetriruumist) võib ahelal kuluda liiga palju aega, et temast tegelikku kasu võiks tõusta. Selle pärast otsitakse, ja leitakse, sellele matemaatiliselt efektiivsemaid alternatiive, millega (st Stan) kasutame nn Hamiltonian Monte Carlo-t.

Hamiltonian Monte Carlo lahendab probleemi jooksutades füüsikalise simulatsiooni käigus kuulikest n -dimnesionaalsel pinnal. See kuulike liigub posteeriori otsinguil kiiremini. Kuulike veedab kauem aega parameetriruumi piirkondades, mis on andmete poolt paremini kinnitatud. Igas punktis randomiseeritakse tema momentum. Pind on miinus log posteerior. See töötab hästi, aga vajab lisaks gradienti e log-posteeriori kurvatuuri, kuulikese massi, ühel trajektooril asuvate sammude hulka ja individuaalsete sammude pikkust. Need lahendatakse automaatselt, aga igaühega neist võivad seonduda veateated ja kvaliteediproblemid.

`n_eff` - effective sample size - sammude arv, arvestades, et puudub autokorrelatsioon ahela järjestikuste sammude vahel.

Chapter 16

Lihtne normaaljaotuse mudel

Kui me eelmises peatükis modeleerisime diskreetseid binaarseid sündmusi (elus või surnud) üle binoomjaotuse, siis edasi tegeleme pidevate suurustega ehk parameetritega, millele saab omistada iga väärtsuse vahemikus -Inf kuni Inf.

```
library(tidyverse)
library(stringr)
library(gapminder)
library(rethinking)
library(bayesplot)
library(gridExtra)
```

Proovime veelkord USA presidentide keskmist pikkust ennustada (sama näide oli bootstrappimisel). Selleks on meil on vaja kahte asja: (1) tõepära mudelite ning (2) igale tõepära mudeli parameetrile oma priorit.

Selline on täismudeli (tõepära ja priorid) struktuur:

```
heights ~ dnorm(mu, sigma) # normal likelihood
mu ~ dnorm(mean = 0, sd = 200) # normal prior for mean
sigma ~ dcauchy(0, 20) #half-cauchy prior for sd
```

Tõepära on siin modeleeritud normaaljaotusena, milles on 2 tuunitavat parameetrit: mu (keskmise) ja sigma (standardhälve). Pelgalt nende kahe parameetri fikseerimine annab meile unikaalse normaaljaotuse. See, et keskmise pikkuse prior on tsentreeritud nullile viib õige pisukesele (nõnda laia priori juures küll pigem märkamatule) mu hinnangu nihkumisele nulli suunas. Selle nihke õigustus on püüd väältida mudeli üle-fittimist ehk teisisõnu ülespoole kallutatud hinnangut keskmisele pikkusele. Sama hästi võiksime kasutada ka priorit `mu ~ dnorm(mean = 178, sd = 10)`, kus 178 on ameerika meeste keskmise pikkuse.

Cauchy prior for sd

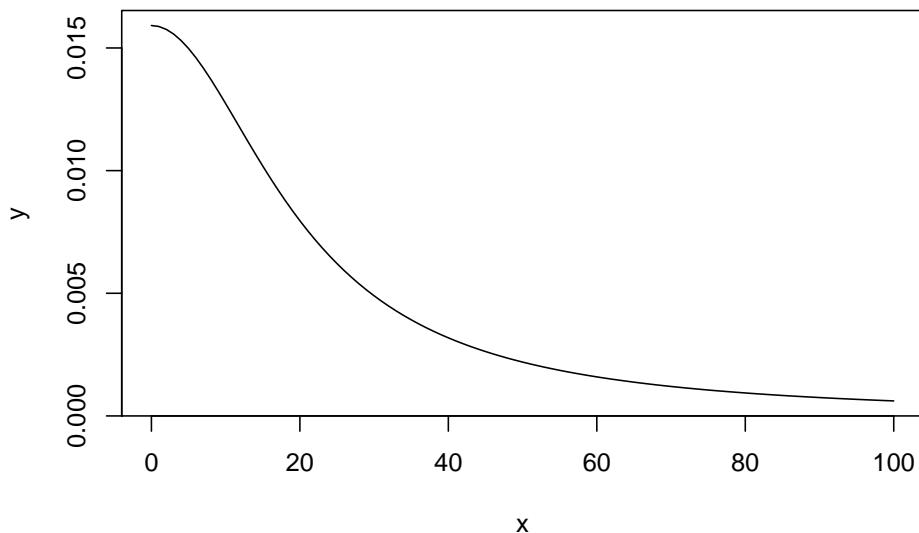


Figure 16.1: Cauchy prior.

Alati tasub mudeli priorid välja plottida, et veenduda, et nad tõesti kajastavad meile taustateadmisi ja on sobivas parameetrvahemikus (bayesi programmeerde default priorid on sageli kas liiga laiad või vastupidi eeldavad, et parameetrväärtused jääävad alla 10 ühiku).

```
x <- 0:100
y <- dcauchy(x, 0, 20)
plot(y ~ x, type = "l" , main = "Cauchy prior for sd")

x <- 150:200
y <- dnorm(x, 0, 200)
plot(y ~ x, type = "l" , main = "Normal prior for mean = 0 and sd = 200")

x <- 150:200
y <- dnorm(x, 178, 10)
plot(y ~ x, type = "l" , main = "Normal prior for mean = 178 and sd = 10")
```

Siin on valida kahe priori vahel mu-le. Võib-olla eelistaksid sina mõnda kolmandat? Kui jah, siis pole muud kui tee valmis ja kasuta!

Sama hästi võiksime tõepära modelleerida ka mõne muu jaotusega (Studenti t jaotus, eksponentsialne jaotus, lognormaaljaotus jne). Sel juhul oleksid meil erinevad parameetrid, mida tuunida, aga põhimõte on sama. Bayes on modulaarne — kui sa põhimõtet tead, pole tehniliselt suurt vahet, millist mudelit

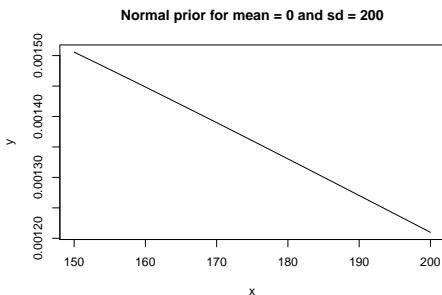


Figure 16.2: Kaks normaaljaotuse priorit.

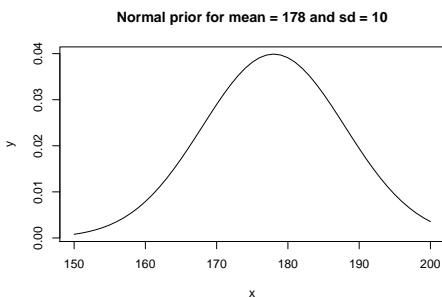


Figure 16.3: Kaks normaaljaotuse priorit.

soovid kasutada.

Näiteks:

```
heights ~ student_t(nu, mu, sigma) # t likelihood
nu ~ dunif(1, 100) # uniform prior for the shape parameter
mu ~ dnorm(mean = 0, sd = 200) # normal prior for mean
sigma ~ dcauchy(0, 20) # half-cauchy prior for sd
```

Normaaljaotusel on kaks parameetrit, millele posteerior arvutada: mu (mean) ja sigma (sd). Seega on vaja ka kahte priorit, üks mu-le ja teine sigma-le. Studenti t jaotuse korral lisandub veel üks parameeter: nu ehk jaotuse kuju määraav parameeter. nu-d saab tuunida 1 ja lõpmatuse vahel. Mida väiksem on nu, seda paksemad tulevad jaotuse sabad. Kui nu on suur, siis on t jaotuse kuju sama, mis normaaljaotusel. Siin andsime nu-le tasase priori 1 ja 100 vahel, hiljem proovime ka teisi prioreid nu-le.

Studenti t jaotus on põnev alternatiiv normaaljaotusele, sest see on vähem tundlik outlieritele. Kuna normaaljaotus langeb servades väga kiiresti siis, kui meil on mõni andmepunkt, mis jäääb jaotuse tipust kaugele, on ainus võimalus selle punkti normaaljaotuse alla mahutamiseks omistada jaotusele väga suur standardhälve. See muudab outlierit sisaldava normaaljaotuse ülemäära laiaks, mis viib analüüsits asjatult kaotatud efektidele. Seevastu t jaotuse sabasid saab

nu abil üles-allä liigutada vastavalt sellele, kas andmed sisaldavad outliereid (selleks tuleb lihtsalt fittida nu parameeter andmete põhjal).

Outlierid toovad meile paksema sabaga jaotuse, mis tipu ümber ei lähe aga kaugeltki nii laiaks, kui samade andmetega fititud normaaljaotus.

Kui lai on meie tõepärafunktsioon?

Normaaljaotusega modelleeritud tõepärafunktsioon on normaaljaotus, mille **keskväärtus** = `mean(valim)` ja mille **standardhälve** = `sd(valim) / sqrt(N)`, kus N on valimi suurus. See tõepärafunktsioon modelleerib meie valimi keskväärtuse kohtamise tõenäosust igal võimalikul parameetrväärtusel. Kui oleme huvitatud USA presidentide keskmisest pikkusest, siis tõepärafunktsioon ütleb iga võimaliku pikkuse kohta, millise tõenäosusega kohtaksime oma valimi keskväärtust juhul, kui just see oleks tegelik presidentide keskmise pikkus. Sigma, mille posteeriori me mudelist arvutame, on aga standardhälve algsete andmepunktide tasemel. See on väga oluline eristus, sest sigma kaudu saab simuerida uusi andmepunkte.

Lihtne või robustne normaalne mudel?

Proovime mudeldada simuleeritud andmete keskväärtust.

```
set.seed(890775)
a <- rnorm(20, mean = 0, sd = 1) # expected mean = 0, sd = 1
b <- c(a, 5, 9) # plus 2 outliers
```

Siin kasutame andmeid, mille keskväärtus on 0.38 ja $sd = 1$ ja millele on lisatud kaks outlierit (5 ja 9). Proovime neid andmeid mudeldada normaaljaotusega tõepäramudeliga ja seejärel üle studenti t jaotuse. Me fitime 4 mudelit, neist 3 koos outlieritega. Mudeli fittimine käib nii, et mcmc ahelad sammuvad parameetriruumis ja iga samm annab meile ühe juhusliku väärtuse posteeriорist. Defaultina on meil üks ahel, mis teeb 1000 sammu (seda saab muuta: `vt ?map2stan`). Kuna ahelad veedavad rohkem aega seal, kus posterioorne tõenäosuspilv on tihedam, siis saab nõnda sämplitud posteeriori juhuvalimi histogrammist posterioorse jaotuse kuju. Veelgi enam, selle asemel, et tegeleda posterioorsete jaotuste matemaatilise analüüsiga (integreerimisega) võime analüüsida oma mcmc sämpelid otse, mis tähendab, et kõrgema matemaatika asemel vajame 2. klassi aritmeetikat.

Kõigepealt ilma outlieriteta mudel normaalse tõepärafunktsiooniga. Me kasutame `sd` priorina pool-Cauchy jaotust, mille tipp on 0 kohal ja millel on piisavalt paks saba suuremate numbrite poole. See on väheinformatiivne prior, mis on nähtud

sd-de puhul mcmc algoritmides hästi töötavat. Andmed võime `map2stan()` funktsiooni sisestada nii listina kui data.frame-na (aga mitte tibble kujul).

```
# Ilma outlierita andmed
m0 <- map2stan(
  alist(
    y ~ dnorm(mu, sigma), # normal likelihood
    mu ~ dnorm(0, 5), # normal prior for mean
    sigma ~ dcauchy(0, 2.5) # half-cauchy prior from sd
  ),
  data = list(y = a))
```

Sama mudel, aga outlieritega andmed. `map2stan()` tõlgib sisestatud mudeli Stan keelde ja see mudel kompileeritakse C++ keelde, milles on kodeeritud Stani mcmc mootor. Kuna kompileerimine on ajakulukas, kasutame m1 fittimiseks rstan raamatukogu (see loetakse sisse rethinkingu dependency-na) ja juba kompileeritud m0 mudelit, millele lisame andmed kahe elemendina: N annab andmete arvu ja y tegelikud andmeväärtused. Selline andmete sisestamise viis on omane Stanile - `map2stan()` arvutab ise kapoti all N-i.

```
m1 <- stan(fit = m0@stanfit,
            data = list(N = length(b),
                        y = b),
            chains = 4)
```

Nüüd studenti t jaotusega töepäramudel. Argumendid cores = 4, chains = 4 tähendavad, et me jooksutame 4 mcmc ahelat kasutades selleks oma arvuti 4 tuuma. Mudeli m2 juures tähendab argument constraints(list(nu = "lower=1")), et mcmc sämpleri ahelad ei lähe kunagi allapoole ühte. See on siin kuna definitsiooni kohaselt ei saa nu olla väiksem kui 1. Argument start annab listi, mis annab iga parameetri jaoks väärtsuse, millega mcmc ahel posteeriori sämplimist alustab. See on vahest vajalik, sest kui mcmc ahelad hakkavad posteeriori töenäosuspilve otsima kaugel selle tegelikust asukohast n-mõõtmelises ruumis (n = mudeli parameetrite arv), siis võib juhtuda, et mudeli fittimine ebaõnnestub ja te saate veateate.

```
m2 <- map2stan(
  alist(
    y ~ student_t(nu, mu, sigma),
    nu ~ dnorm(5, 10),
    mu ~ dnorm(0, 5),
    sigma ~ dcauchy(0, 2.5)
  ),
  data = list(y = b),
  constraints = list(nu = "lower=1"),
  start = list(mu = mean(b), sigma = sd(b), nu = 10),
  cores = 4,
  chains = 4
```

```
)
m2 <- readRDS("data/stan_m2.rds")
m2@stanfit
#> Inference for Stan model: y ~ student_t(nu, mu, sigma).
#> 4 chains, each with iter=2000; warmup=1000; thin=1;
#> post-warmup draws per chain=1000, total post-warmup draws=4000.
#>
#>      mean se_mean    sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
#> mu     0.28    0.00 0.22 -0.11  0.14  0.27  0.41  0.76 2066   1
#> sigma   0.78    0.01 0.27  0.39  0.59  0.74  0.92  1.40 1254   1
#> nu      2.21    0.04 1.48  1.04  1.39  1.84  2.53  5.57 1644   1
#> dev     78.68   0.10 3.35 74.84  76.19  77.81  80.25 87.47 1057   1
#> lp__   -27.23   0.04 1.45 -31.06 -27.86 -26.88 -26.16 -25.55 1323   1
#>
#> Samples were drawn using NUTS(diag_e) at Mon Oct 23 15:51:33 2017.
#> For each parameter, n_eff is a crude measure of effective sample size,
#> and Rhat is the potential scale reduction factor on split chains (at
#> convergence, Rhat=1).
```

Ja viimasena studenti t mudel, kus nu on fikseeritud konstandina. Kuna me ei fiti nu-d mudeli parameetrina, pole meil vaja ka priorit nu-le. Me teeme selle mudeli, sest nu täpsel väärtsel pole väga suurt mõju tulemustele. Me lihtsalt fikseerime nu suvalisele väärtsusele, mis annab t jaotusele piisavalt paksud sabad.

```
m3 <- map2stan(
  alist(
    y ~ student_t(4, mu, sigma),
    mu ~ dnorm(0, 5),
    sigma ~ dcauchy(0, 2.5)
  ),
  data = list(y = b),
  constraints = list(nu = "lower=1"),
  start = list(mu = mean(b), sigma = sd(b)),
  cores = 4,
  chains = 4)
```

Üks esimesi asju mida koos parameetrite vaatamisega teha on lisaks vaadata, kas ka ahelad konvergeerusid. Selleks saab mugavalt kasutada `rethinking::tracerplot()` funktsiooni.

```
tracerplot(m2)
```

Pildilt on näha, et neli ahelat (4 värvi) on hästi konvergeerunud. Hall ala on nn warmup ala, mille tulemusi ei salvestata. Muidu astub iga ahel sammu kaupa ja iga edukas samm salvestatakse ühe posteeriori väärtsusena. Ahel sämplib korraga mu, sigma ja nu väärtsusi n-mõõtmelises ruumis ($n = \text{mudeli parameetrite arv}$),

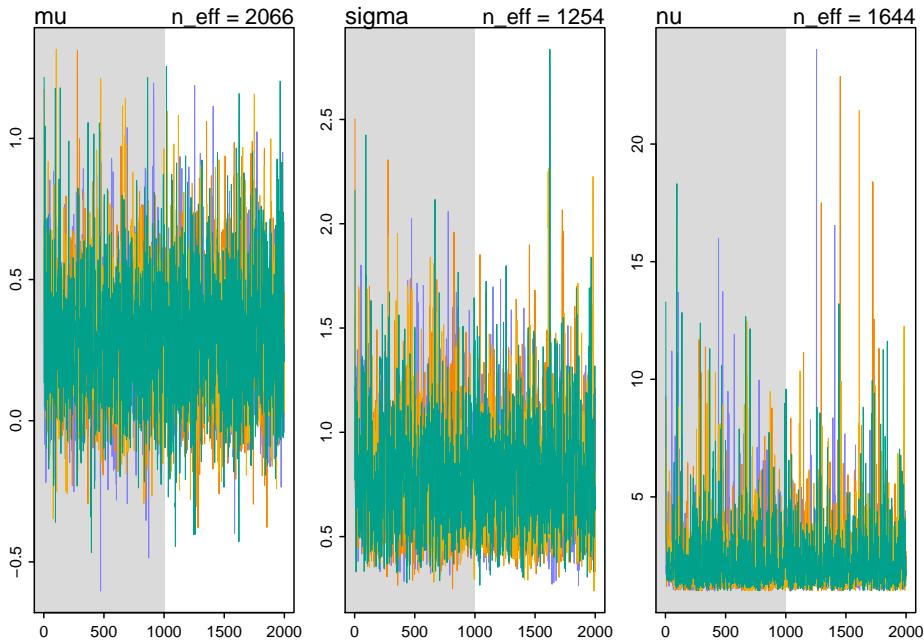


Figure 16.4: Traceplot markovi ahelate inspekteerimiseks.

mis tähendab, et ahela iga samm salvestatakse n kõrvuti numbrina.

Kui näit sigma kõrgema väärtsusega kaasneb keskeltäbi kõrgem (või madalam) mu väärtsus, on sigma ja mu omavahel korreleeritud. Et kontrollida parameetrite posterioorsete väärustuste korrelatsioone kasutame funktsiooni `rthinking::pairs()`:

```
pairs(m2)
```

Normaaljaotus on selle poolest eriline, et tema parameetrid mu ja sigma ei ole korreleeritud. Paljud teised mudelid ei ole nii lahked. Siin on meil mõõdukas korrelatsioon nu ja sigma vahel. See on igati loogiline ja ei häiri meid.

MCMC ahelate kvaliteet

Kui Rhat on 1, siis see tähendab, et MCMC ahelad on ilusti jooksnud ja posteeriori sämplinud. Kui Rhat > 1.1 , siis on probleem. Suur Rhat viitab, et ahel(ad) pole jõudnud konvergeeruda. Kui ahelad ei konvergeeru, siis võib karta, et nad ei sämpli ka sama posteeriori jaotust. Kontrolli, kas mudeli kood ei sisalda vigu. Kui ei, siis vahest aitab, kui pikendada warm-up perioodi (`map2stan(..., iter = 3000, warmup = 2000)` pikendab `default` warm-upi 2 korda). Vahest

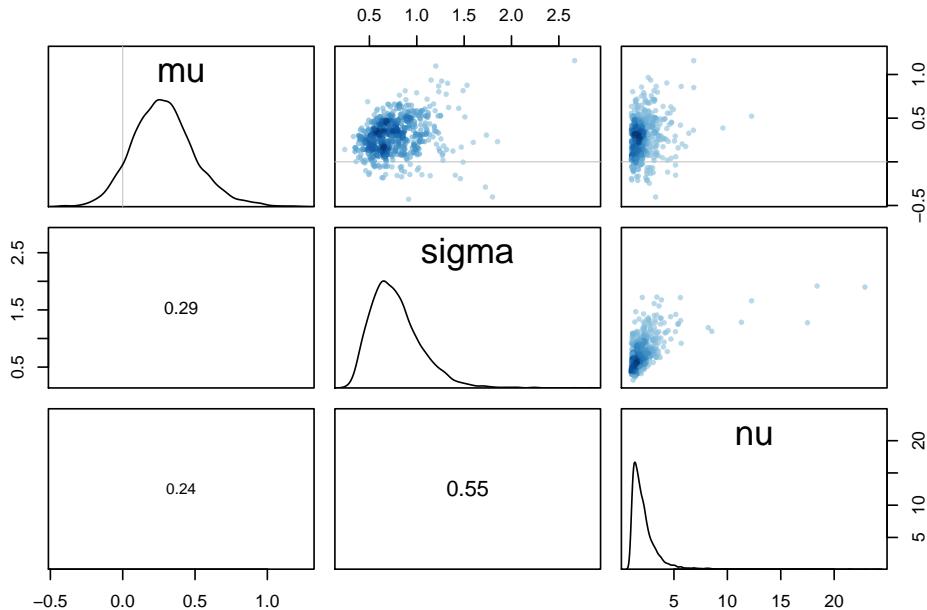


Figure 16.5: Korrelatsiooniplot mudeli parameetritele.

aitab mudeli re-parametriserimine (siin on lihtne trikk tekitada priorid, mis ei erineks väga palju oma vahemiku poolest; sellega kaasneb sageli andmete tsentreerimine või standardiseerimine; vt allpool).

`n_eff` on efektiivne valimi suurus, mis hindab iseseisvalt sümplitud andmete arvu ning see ei tohi olla väga väike. Kui `n_eff` on palju väiksem kui jooksutatud markovi ahela pikkus (iga ahel on defaultina 1000 iteratsiooni pikk), on ahel jooksnud ebaefektiivselt. See ei tähenda tingimata, et posteerior vale oleks. Reegilina peaks $\text{Neff}/N > 0.1$

Ahelas peavad plotitud kujul välja nägema nagu karvased töugud, mis on ilma paljaste laikudeta. Kui ahelad omavad pikki sirgeid lõike (`n_eff` tuleb siis väga madal), kus ahel ei ole töötanud, siis see rikub korralikult posteeriori. Tüüpiliselt aitavad nõrgalt informatiivsed priorid — priorite õige valik on sama palju arvutuslik vajadus kui taustainfo lisamine. Igal juhul tuleb vältida aladefineeritud tasaseid prioreid, mis võimaldavad ahelatel sämplida lõpmatust ja sel viisil õige tee kaotada. Peale selle, tasased priorid, mis ütlevad, et kõik parameetri väärtsused on võrdsest tõenäolised, kajastavad harva meie tegelikke taustateadmisi.

Halvad WARNING-ud: divergent transitions (too many), BMFI too low — võivad tähendada, et ahelad ei tööta korralikult. WARNING-ute kohta saad abi siit <http://mc-stan.org/misc/warnings.html>.

Ilusamad parameetriplotid saab kasutades “bayesplot” raamatukogu funktsioone.

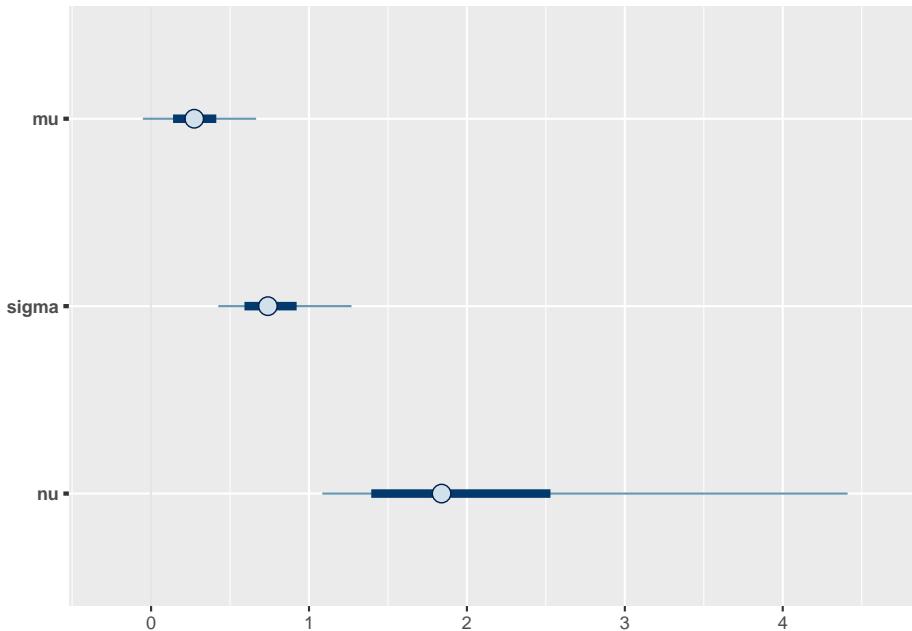


Figure 16.6: Posterior CI plot.

Esiteks usalduspiirid:

```
fit2d <- as.data.frame(m2@stanfit)
pars <- names(fit2d)

# inner interval = 50% CI and outer interval = 95% CI.
mcmc_intervals(fit2d,
  pars = pars[1:3],
  prob = 0.5,
  prob_outer = 0.90)
```

Ja teiseks täis posteriorid.

```
mcmc_areas(fit2d, pars = pars[1:2], prob = 0.8)
```

Funktsiooniga `rethinking::extract.samples()` saame koos sämplitud parameetrite numbrid kõrvuti (rea kaupa) tabelisse.

```
m2sampl <- extract.samples(m2) %>%
  as.data.frame() %>%
  mutate(CV = sigma / mu)
```

Sellest tabelist võib arvutada posterioreid ka uutele “väljamõeldud”

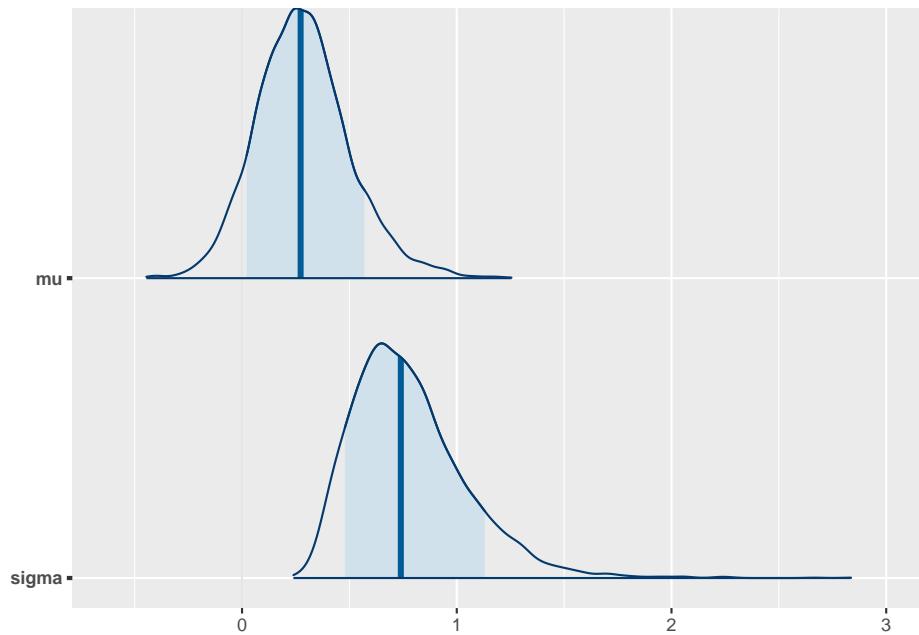


Figure 16.7: Posteriorite tihedusplot.

parameetritele. Näiteks arvutame posteriori CV-le:

```
ggplot(m2sampl, aes(CV)) +
  geom_density(breaks = seq(0, 1, by = 0.1)) +
  xlim(0, 10)
#> Warning: Ignoring unknown parameters: breaks
#> Warning: Removed 609 rows containing non-finite values (stat_density).
```

Kuna posterior iseloomustab meie teadmiste piire, siis võime selle abil küsida, kui suure töenäosusega jäääb tõeline CV näiteks parameetrivahemikku 2 kuni 5?

```
intv <- filter(m2sampl, between(CV, 2, 5)) %>% nrow(.) / nrow(m2sampl)
intv
#> [1] 0.415
```

Vastus on, et me arvame 42 kindlusega, et tõde jäääb kuskile sellesse vahemikku.

Võime ka küsida, millesesse vahemikku jäääb näiteks 67% meie usust mu tõelise väärvtuse kohta?

```
HPDI(m2sampl$CV, prob = 0.67)
#> 10.67 0.671
#> 0.964 4.058
```

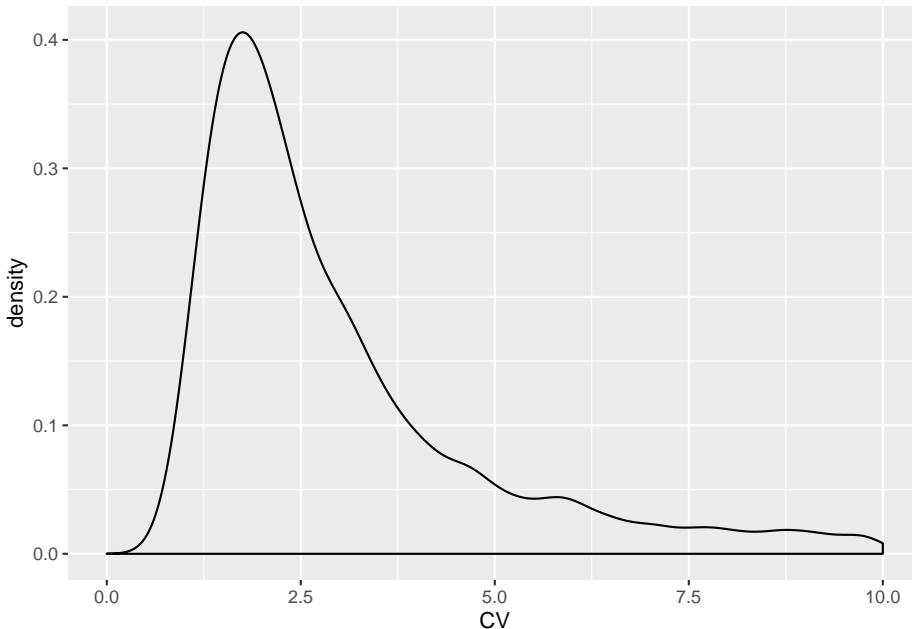


Figure 16.8: Posterior uuele parameetrile

Nüüd võrdleme nelja fititud mudelit, et otsustada, milline mudel kirjeldab kõige paremini outlieritega andmeid. m_0 on ilma outlierita mudel ja me tahame teada, milline mudel m_1 , m_2 või m_3 annab sellele kõige lähedasemad tulemused.

```
coeftab_plot(coeftab(m0, m1, m2, m3),
             pars = c("mu", "sigma"),
             prob = 0.5)
```

Me sättisime usalduspiiriid 0.5 peale, mis tähendab, et need ennustavad, kuhu peaks mudeli järgi jäma parameetri tegelik väärthus 50%-se töenäosusega. Nagu näha, on m_2 ja m_3 posteriorid palju lähemal m_0 -le kui normaaljaotusega fititud m_1 oma. Eriti drastilised on erinevused sigma hinnangule. Lisaks, m_1 mudeli mu usaldusintervall on palju laiem kui m_0 , m_2 ja m_3 oma — mudel nagu saaks aru, et andmed lõhnavad kala järgi.

Näide: USA presidentide keskmine pikkus

Läheme tagasi normaaljatuse ja USA presidentide juurde. Kõigepealt defineerime priorid. Alati on mõistlik priorid välja joonistada ja vaadata, kas nad vastavad meie ootustele. Pea meeles, et sigma ehk σ on samades ühikutes, mis mõõtmisandmed.

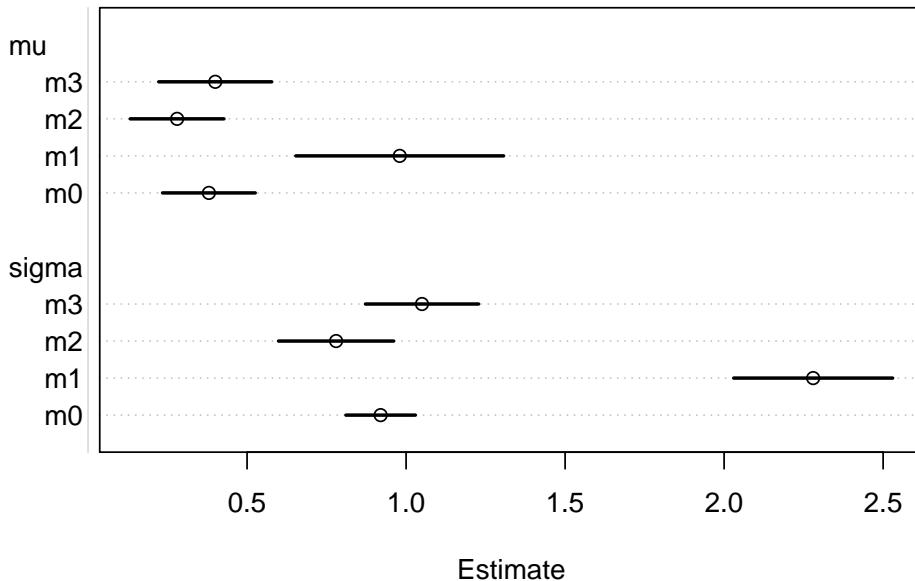


Figure 16.9: Võrdlev plot mitme mudeli posteerioritele.

Kui sulle need priorid ei meeldi, tuuni priorite parameetreid ja proovi uuesti plottida.

```
x <- -500:500
y <- dnorm(x, 0, 200)
plot(x, y, main = "Prior for mu", type = "l")
```

Siin kasutame nõrgalt informatiivseid prioreid. Idee on selles, et normaaljaotus, mis on tsentreeritud 0 ümber, tõmbab meie posteeriorit nõrgalt null poole (nõrgalt, sest jaotus on hästi lai võrreldes töepärafunktsiooniga). Pane tähele, et oma priori kohaselt usume me, et 50% tõenäususega on USA presidentide keskmise pikkus negatiivne. See prior on tehniline abivahend, mitte meie tegelike uskumuste peegeldus presidentide kohta. Aga tehniliselt kõik töötab selles mõttes, et andmed domineerivad posteeriori üle ja priori sisuliselt ainus ülesanne on veidi MCMC mootori tööd lihtsustada.

Sigma priorina kasutame half-Cauchy jaotust, mis on samuti väheinformatiivne. Half-Cauchy ei saa olla < 0 ja on meile soodsa kujuga sest annab suurema tõenäosuse nullile lähemal asuvatele sd-väärtustele — aga samas, kuna ta on paksu sabaga, ei välista see ka päris suuri sd väärtusi.

```
x <- 0:200
y <- dcauchy(x, 0, 10)
plot(x, y, main = "Prior for sigma", type = "l")
```

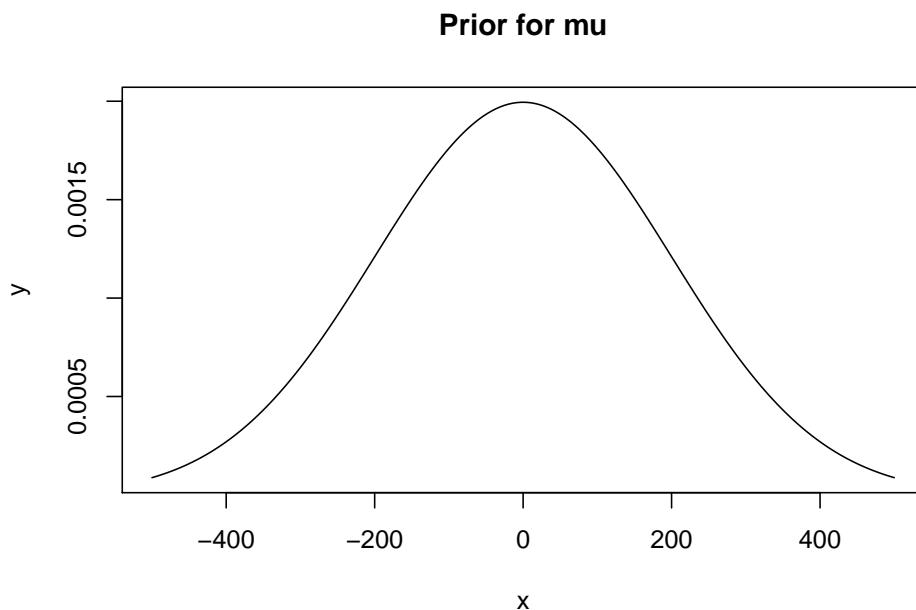


Figure 16.10: Prior keskmisele.

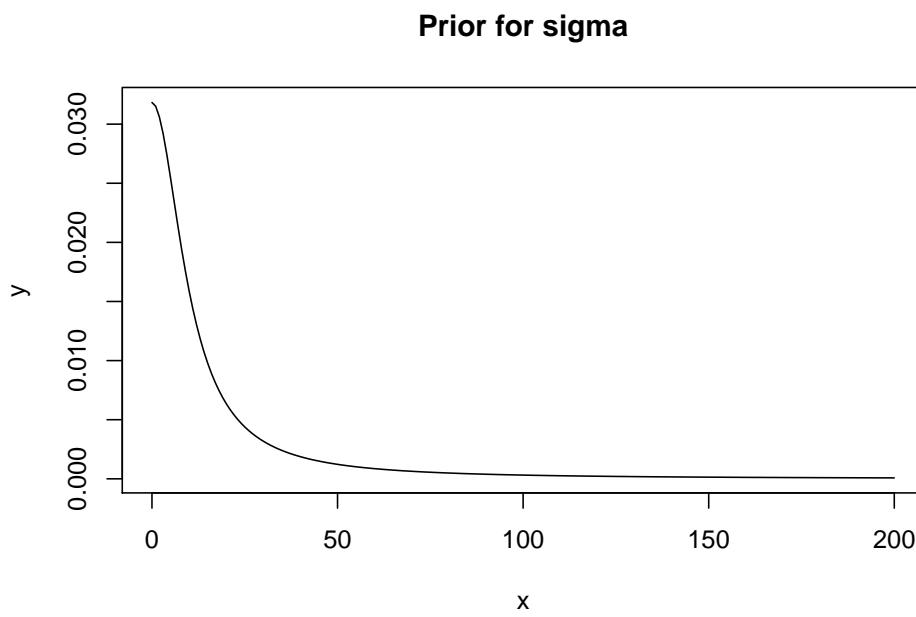


Figure 16.11: Prior SD-le

Tekitame andmeraami analüüsiks ja mudeli, mis põhineb normaalsel tõepära-funktsoonil.

```
heights <- c(183, 192, 182, 183, 177, 185, 188, 188, 182, 185)
us_presidents <- data_frame(Height = heights, id = "usa")
potusm1 <- map2stan(
  alist(
    Height ~ dnorm(mu, sigma), # normal likelihood
    mu ~ dnorm(0, 200), # normal prior for mean
    sigma ~ dcauchy(0, 10) # half-cauchy prior from sd
  ), data = us_presidents
)
```

Mudeli koefitsiendid:

```
precis(potusm1)
#>      Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> mu     184.5     1.5    181.90    186.71    482     1
#> sigma   4.7     1.3     2.87     6.36    471     1
```

Nüüd teeme katse võrrelda USA presidentide ja Euroopa ning mujalt pärit riigijuhtide keskmisi pikkusi. Kõigepealt loome analüüsitava andmeraami.

```
world_leaders <- read_csv2("data/world_leaders.csv")
presidents <- world_leaders %>%
  select(Country, Height) %>%
  bind_rows(us_presidents)
knitr::kable(head(presidents))
```

Country	Height
Canada	188
Cuba	190
France	170
France	165
France	189
France	172

Ja siin on mudel. Nüüd on mu ümber defineeritud kui mu1[indeks], mis tähendab, et mu1 saab kaks hulka väärtsusi, üks kummagil indeks muutuja tasemel. Sellega jagame oma andmed kahte ossa (USA versus Euroopa ja muu maailm), mida analüüsime eraldi. Sigma on mõlemale kontinendile sama, mis tähendab, et mudel eeldab, et presidentide pikkuste jaotus on mõlemal kontinendl identne.

```
# Split into 2 groups
presidents <- presidents %>%
  mutate(Groups = case_when(
    Country == "USA" ~ "USA",
    Country != "USA" ~ "World"
  ))
```

Adult human height varies country-by-country, we take 170 cm as relatively safe prior for male height.

```
potusm2 <- map2stan(
  alist(
    Height ~ dnorm(mu, sigma),
    mu <- mu_1[Groups], # mu is redefined as mu_1, which takes values at each indeks level
    mu_1[Groups] ~ dnorm(170, 10), # normal prior for mean
    sigma ~ dcauchy(0, 10) # half-cauchy prior from sd
  ),
  data = presidents)

precis(potusm2, depth = 2)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> mu_1[1] 182.8   3.57    177.11     188.2   885     1
#> mu_1[2] 176.3   1.85    173.33     179.2   836     1
#> sigma     11.7   1.26     9.86      13.8   628     1
```

Me võime ka vaadata 2 gruopi standardhälbeid lahus. Järgnevas mudelis on mõistlik ahelale stardipositsioon ette anda.

```
## Calculate start values
startvalues <- presidents %>%
  group_by(Groups) %>%
  summarise_at(vars(Height), funs(mean, sd))
## Fit model
potusm2.1 <- map2stan(
  alist(
    Height ~ dnorm(mu, sigma),
    mu <- mu_1[Groups],
    sigma <- sigma_1[Groups],
    mu_1[Groups] ~ dnorm(170, 10), # normal prior for mean
    sigma_1[Groups] ~ dcauchy(0, 10) # half-cauchy prior from sd
  ),
  data = presidents,
  start = list(mu_1 = startvalues$mean,
               sigma_1 = startvalues$sd)
)

tracerplot(potusm2.1, n_cols = 2)
```

Tulemus ES-i osas tuleb üsna sarnane.

```
plot(coeftab(potusm2, potusm2.1))

precis(potusm2, depth = 2)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> mu_1[1] 182.8   3.57    177.11     188.2   885     1
```

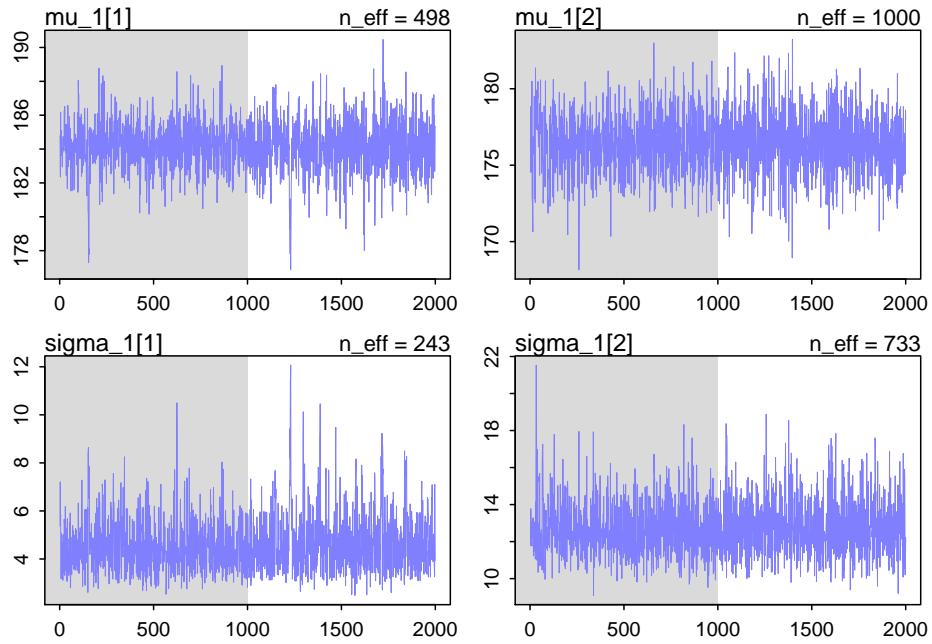


Figure 16.12: Traceplot.

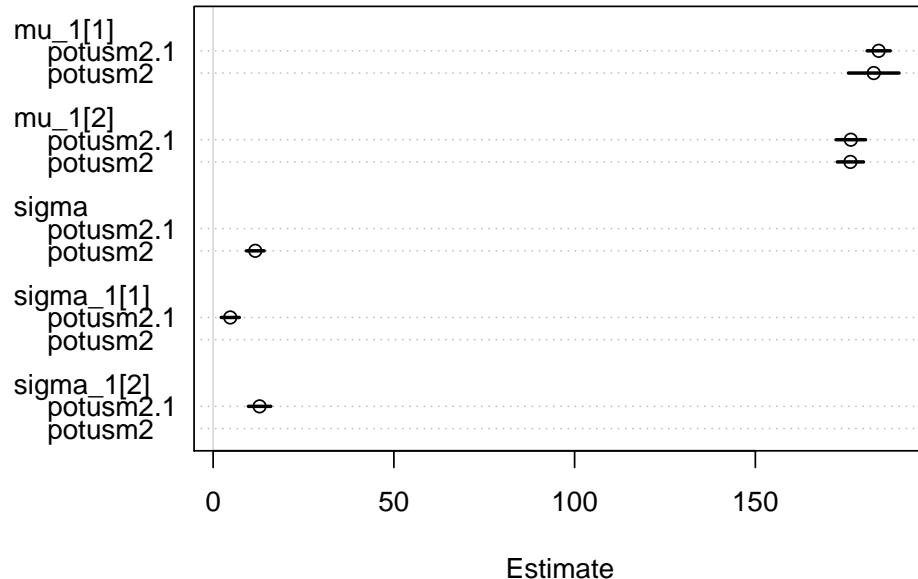


Figure 16.13: mudelite võrdlusplot.

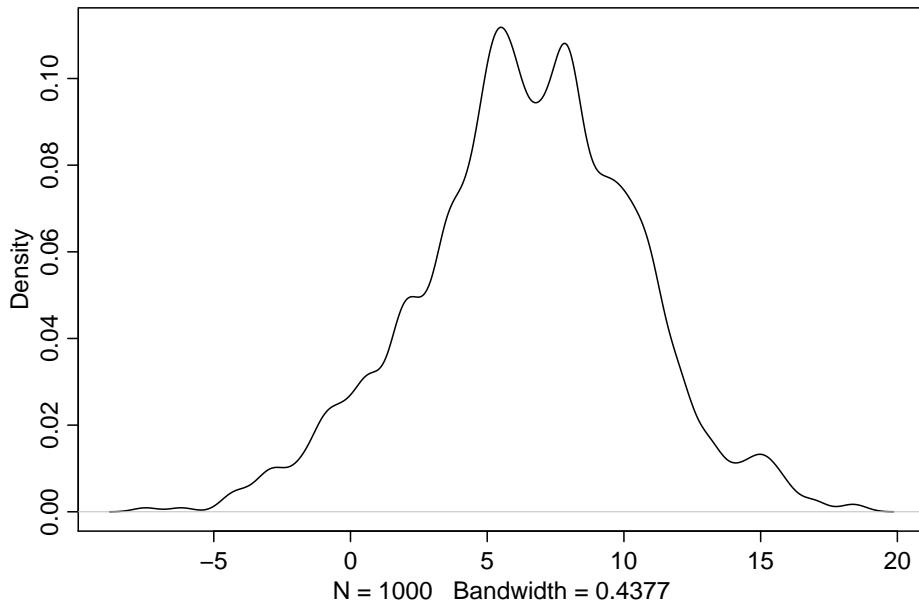


Figure 16.14: Posterior ES-le.

```
#> mu_1[2] 176.3 1.85 173.33 179.2 836 1
#> sigma     11.7 1.26 9.86 13.8 628 1
```

Siin tuleb kasulik trikk: me lahutame rea kaupa mu1[1] posteriori sampli liikmed mu1[2] sampli liikmetest. Nii saame posteriori efekti suurusele ehk hinnangu sellele, mitme cm võrra on USA presidendid keskmiselt pikemad kui Euroopa omad!

```
samplespm2 <- extract.samples(potusm2) %>%
  as.data.frame() %>%
  mutate(ES = mu_1.1 - mu_1.2)
dens(samplespm2$ES)

median(samplespm2$ES)
#> [1] 6.56
rethinking::HPDI(samplespm2$ES, prob = 0.9)
#> 10.9 0.9/
#> -1.09 12.02
mean(samplespm2$ES < 0)
#> [1] 0.065
```

Võrdse SD-ga mudeli järgi on USA presidendid keskelt läbi 6.6 cm pikemad, ebakindlus selle hinnangu ümber on suur – 90% HDI on -1.1 kuni 12 ja tõenäosus

et pikkuste erinevus on väiksem kui 0 on 0.06.

```
samplesm2.1 <- extract.samples(potusm2.1) %>%
  as.data.frame() %>%
  mutate(ES = mu_1.1 - mu_1.2)
median(samplesm2.1$ES)
#> [1] 7.54
HPDI(samplesm2.1$ES, prob = 0.9)
#> 10.9 0.91
#> 3.39 12.16
mean(samplesm2.1$ES < 0)
#> [1] 0.001
```

Erineva SD-ga mudeli järgi on riigijuhtide pikkuste vahe 7.5 cm, ebakindlus väiksem – 90% HDI on 3.4 kuni 12.2 ja töenäosus et pikkuste erinevus on väiksem kui 0 on 0.

See ei tähenda tingimata, et me peaksime eelistama teist mudelit. Oluline on, mida me teoreetiliselt usume, kas seda, et tegelik presidentide varieeruvus on USAs ja Euroopas võrdne, või mitte.

Lineaarne regressioon

Eelmises peatükis hindasime ühe andmekogu (näiteks mõõdetud pikkuste) põhjal ehitatud mudelite parameetreid (näiteks keskmist ja standardhälvet). Nüüd astume sammu edasi ja hindame kahe muutuja (näiteks pikkuse ja kaalu) koosvarieeruvust. Selleks ehitame mudeli, mis sisaldab mõlemaid muutujaid ja küsimuse: kui palju sõltub y varieeruvus x varieeruvusest. Lihtsaim viis sellele küsimusele läheneda on lineaarse regressiooni kaudu. Me ehitame lineaarse mudeli, mis vaatab kaalu-pikkuse paare (igal subjektil mõõdeti kaal ja pikkus ning mudel vaatab kaalu ja pikkuse koos-varieeruvust subjektide vahel). Enam ei tohiks tulla üllatusena, et meie arvutused ei anna numbrilist hinnangut mitte teaduslikule küsimusele selle kohta kuidas y -i väärtsused sõltuvad x -i väärustest, vaid mudeli parameetritele. Meie mudel on sirge võrrand $y = a + b * x$ ja tavapärases R-i notatsioonis kirjutatakse see $y \sim x$.

Kuna pikkused ja kaalud on igavad, proovime vaadata kuidas riigi keskmise eluga on seotud riigi rikkusega.

lm() - vähimruutude meetodiga fititud lineaarsed mudelid

Kautame gapminder andmeid aastast 2007.

```
library(gapminder)
# Select only data from year 2007
g2007 <- gapminder %>% filter(year == 2007)
knitr::kable(head(g2007))
```

country	continent	year	lifeExp	pop	gdpPercap
Afghanistan	Asia	2007	43.8	31889923	975
Albania	Europe	2007	76.4	3600523	5937
Algeria	Africa	2007	72.3	33333216	6223
Angola	Africa	2007	42.7	12420476	4797
Argentina	Americas	2007	75.3	40301927	12779
Australia	Oceania	2007	81.2	20434176	34435

Enne kui SKP ja eluea seoseid otsima hakkame, vaatame, mis juhtub, kui me arvutame ainult interceptiga mudeli, kus puudub SKP (kasutades lihtsuse mõttes mudeli fittimiseks nn vähimruutude meetodit `lm()` funktsooni abil).

```
gapmod1 <- lm(lifeExp ~ 1, data = g2007)
summary(gapmod1)
#>
#> Call:
#> lm(formula = lifeExp ~ 1, data = g2007)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -27.39  -9.85   4.93   9.41  15.60
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  67.01      1.01   66.1   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 12.1 on 141 degrees of freedom
```

Ok, intercept = 67. Mida see tähendab?

```
mean(g2007$lifeExp)
#> [1] 67
```

See on lihtsalt parameetri, mida me ennustame, keskmise väärustus ehk keskmise eluiga üle kõikide riikide.

Nüüd fitime mudeli, kus on olemas SKP ja eluea seos aga puudub lõikepunkt.

```
gapmod2 <- lm(lifeExp ~ 0 + gdpPercap, data = g2007)
summary(gapmod2)
#>
```

```

#> Call:
#> lm(formula = lifeExp ~ 0 + gdpPercap, data = g2007)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -65.5   17.6   44.9   54.7   67.0
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> gdpPercap 0.002951   0.000218   13.5   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 45.1 on 141 degrees of freedom
#> Multiple R-squared:  0.565, Adjusted R-squared:  0.562
#> F-statistic: 183 on 1 and 141 DF, p-value: <2e-16

p <- ggplot(g2007, aes(gdpPercap, lifeExp)) +
  geom_point() +
  geom_line(aes(y = .fitted), data = gapmod2)
p

```

Nüüd on intercept surutud väärustusele $y = 0$.

Ja lõpuks täismudel

```

gapmod3 <- lm(lifeExp ~ gdpPercap, data = g2007)
summary(gapmod3)
#>
#> Call:
#> lm(formula = lifeExp ~ gdpPercap, data = g2007)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -22.83  -6.32   1.92   6.90  13.13
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 5.96e+01  1.01e+00   59.0   <2e-16 ***
#> gdpPercap   6.37e-04  5.83e-05   10.9   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 8.9 on 140 degrees of freedom
#> Multiple R-squared:  0.461, Adjusted R-squared:  0.457
#> F-statistic: 120 on 1 and 140 DF, p-value: <2e-16

```

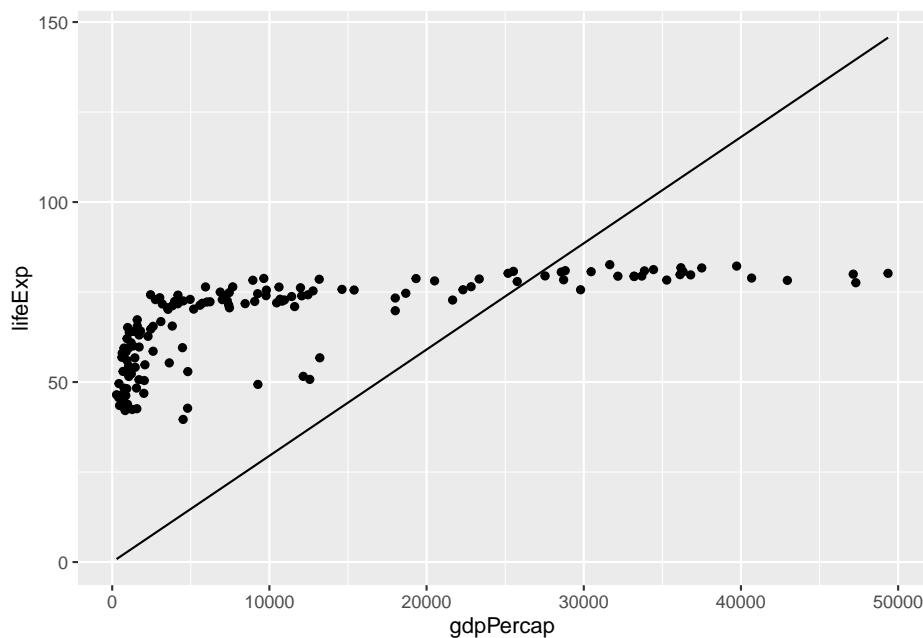


Figure 16.15: Nulli surutud interceptiga lineaarne regressioon eluea sõltuvusele SKP-st.

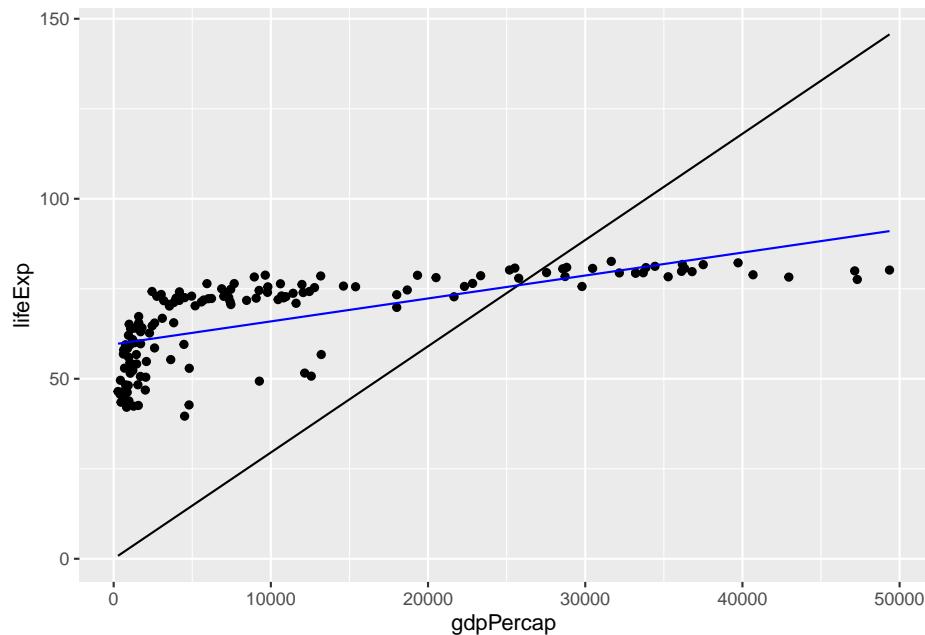


Figure 16.16: Täismudeliga regressioon.

```
p + geom_line(aes(y = .fitted), data = gapmod3, color = "blue")
```

Kuidas me seda m3 mudelit tõlgendame? Esiteks, Intercept on 59.6, mis tähistab, et mudel ennustab, et kui riigi SKP = 0 USD, siis selle riigi elanike keskmise eluiga on ligi 60 aastat. See on selgelt imelik, sest ühegi riigi SKP ei ole null, ja kui oleks, oleks seal ka eluiga 0 (selle järgi peaksime eelistama mudelit gapmod2, kus me oleme intercepti nulli surunud).

Teiseks, koefitsient $b = 0.00064$, mis on üsna väike arv. See tähistab, et SKP tõus 1 USD võrra tõstab eluiga keskmiselt 0.00064 aasta võrra (ja SKP tõus 1000 USD võrra tõstab eluiga 0.64 aasta võrra). Muidugi ainult siis, kui uskuda mudelit.

Kolmandaks, adjusted R squared on 0.46, mis tähistab et mudeli järgi seletab SKP varieerumine 46% eluea varieeruvusest riikide vahel.

Hea küll, aga milline mudel on siis parim?

```
knitr::kable(AIC(gapmod1, gapmod2, gapmod3))
```

	df	AIC
gapmod1	2	1113
gapmod2	2	1487
gapmod3	3	1028

AIC on *Aikakee informatsiooni kriteerium*, mis võtab arvesse nii mudeli fiti headuse kui mudeli parameetrite arvu. Kuna R saab parameetreid lisades ainult kasvada ja me teame, et mingist hetkest oleme niikuinii oma mudeli üle fittinud, siis otsime AIC-i abil kompromissi: võimalult hea fit võimalikult väikese parameetrite arvuga. AIC on suhteline mõõt, selle absoluutnäit ei oma mingit tähdust. Me eelistame väiksema AIC-ga mudelit nende mudelite seast, mida me võrdleme. See ei tähenda, et võitnud mudel oleks hea mudel — alati on võimalik, et kõik head mudelid jäid võrdlusest välja.

Seega parim mudel on gapmod3 ja kõige kehvem on gapmod2, mille lõikepunkt on realistlikult nulli fikseeritud!

Chapter 17

Bayesi meetodil lineaarse mudeli fittimine

```
library(tidyverse)
library(gapminder)
library(rethinking)
```

Nüüd Bayesi mudelid. “rethinking” paketi `glimmer()` on abivahend, mis konverteerib `lm()` mudeli kirjelduse Bayesi mudeli kirjelduseks kasutades normaaljaotusega töepära mudelit. Intercept only model

```
g2007 <- gapminder %>%
  filter(year == 2007)
intercept_only <- glimmer(lifeExp ~ 1, data = g2007)
#> alist(
#>   lifeExp ~ dnorm(mu, sigma),
#>   mu <- Intercept,
#>   Intercept ~ dnorm(0, 10),
#>   sigma ~ dcauchy(0, 2)
#> )
```

Ainult interceptiga mudel. Keskväärtus ehk `mu` on ümber defineeritud kui intercept, aga see annab talle lihtsalt uue nime. Sama hästi oleksime võinud fittida mudelit, kus hindame otse `mu` keskväärtust (nagu me eelmises peatükis tegime). Pane tähele, et võrreldes `lm()` funktsiooniga on meil mudelis lisaparameeter — `sigma`. Kui `Intercept` annab meile keskmise eluea, siis `sigma` annab eluigade standardhälbe riikide vahel.

Kui me tahame fittida lineaarset mudelit, siis peab töepära funktsioon olema kas normaaljaotus või studenti t jaotus.

194 CHAPTER 17. BAYESI MEETODIL LINEAARSE MUDELI FITTIMINE

```
gapmod4 <- map2stan(flist = intercept_only$f, data = intercept_only$d)

precis(gapmod4)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> Intercept 66.3   0.99     64.6     67.8   694     1
#> sigma      12.1   0.71     11.0     13.2   554     1
```

Nüüd ilma interceptita mudel

```
no_intercept <- glimmer(lifeExp ~ -1 + gdpPercap, data = g2007)
#> alist(
#>   lifeExp ~ dnorm(mu , sigma ),
#>   mu <- b_gdpPercap*gdpPercap,
#>   b_gdpPercap ~ dnorm(0,10),
#>   sigma ~ dcauchy(0,2)
#> )
```

Selline Bayesi mudeli esitus on “ilusam” kui lm() sest ta toob mudeli eksplitsiitselt välja (samas kui lm notatsioon ütleb, et mudel on “miinus intercept”)

```
gapmod5 <- map2stan(flist = no_intercept$f, data = no_intercept$d)

precis(gapmod5)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> b_gdpPercap 0.0   0.00     0.0     0.0   888     1
#> sigma        45.2  2.75     40.8    49.4   187     1
```

Ja lõpuks täismudel:

```
full_model <- glimmer(lifeExp ~ gdpPercap, data = g2007)
#> alist(
#>   lifeExp ~ dnorm(mu , sigma ),
#>   mu <- Intercept +
#>         b_gdpPercap*gdpPercap,
#>   Intercept ~ dnorm(0,10),
#>   b_gdpPercap ~ dnorm(0,10),
#>   sigma ~ dcauchy(0,2)
#> )

gapmod6 <- map2stan(flist = full_model$f, data = full_model$d)

compare(gapmod4, gapmod5, gapmod6)
#>           WAIC pWAIC dWAIC weight    SE    dSE
#> gapmod6 1028   2.6   0.0     1 14.07   NA
#> gapmod4 1113   1.5  85.7     0 12.09  9.67
#> gapmod5 1486   0.8 458.8     0  7.29 15.70
```

Jäalle on täismudel võitja ja kui intercept nulli suruda, saame kehveima tulemuse. Siin me kasutame AIC-i Bayesi analoogi WAIC, mis nende mudelite peal peaks

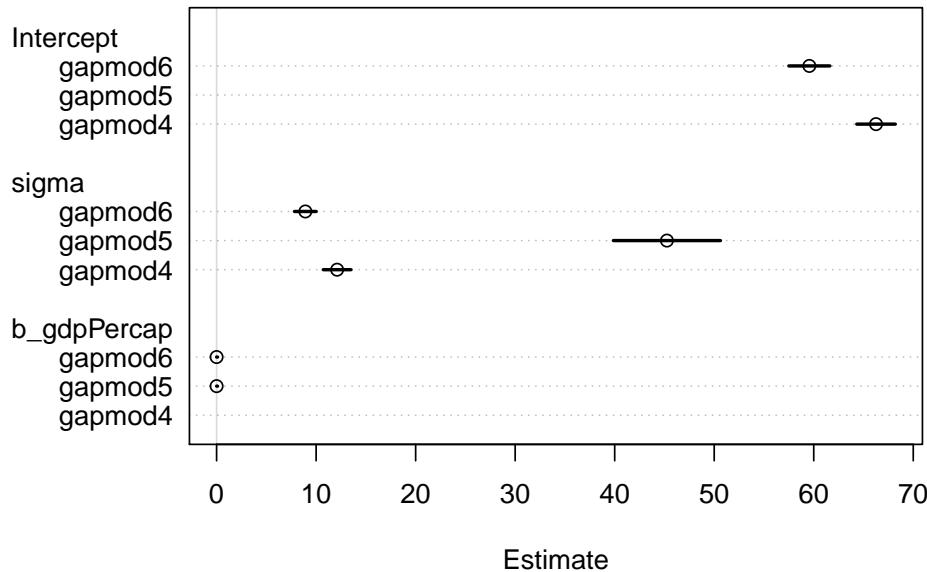


Figure 17.1: Mudelite võrdlusplot.

töötama veidi paremini kui AIC. Aga see on tehniline detail. WAIC abil mudeleid võrreldes saame muuhulgas mudeli kaalu. Antud juhul on 100% kaalust gapmo6-l ja ülejäänud mudelitele ei jäää midagi.

```
plot(coeftab(gapmod4, gapmod5, gapmod6))
```

Viime SKP andmed log-skaalasse ja proovime uuesti. See tähendab, et me arvame, et iga SKP kümnekordne tõus võiks kaasa tuua eluea tõusu x aasta võrra.

```
g2007 <- g2007 %>%
  mutate(l_GDP = log10(gdpPercap))
# glimmer(lifeExp ~ -1 + l_GDP, data = g2007)
gapmod7 <- map2stan(alist(
  lifeExp ~ dnorm(mu, sigma),
  mu <- b_gdp * l_GDP,
  b_gdp ~ dnorm(0, 10),
  sigma ~ dcauchy(0, 2)
), data = g2007)

gapmod8 <- map2stan(alist(
  lifeExp ~ dnorm(mu, sigma),
  mu <- Intercept + b_gdp * l_GDP,
  Intercept ~ dnorm(0, 100),
```

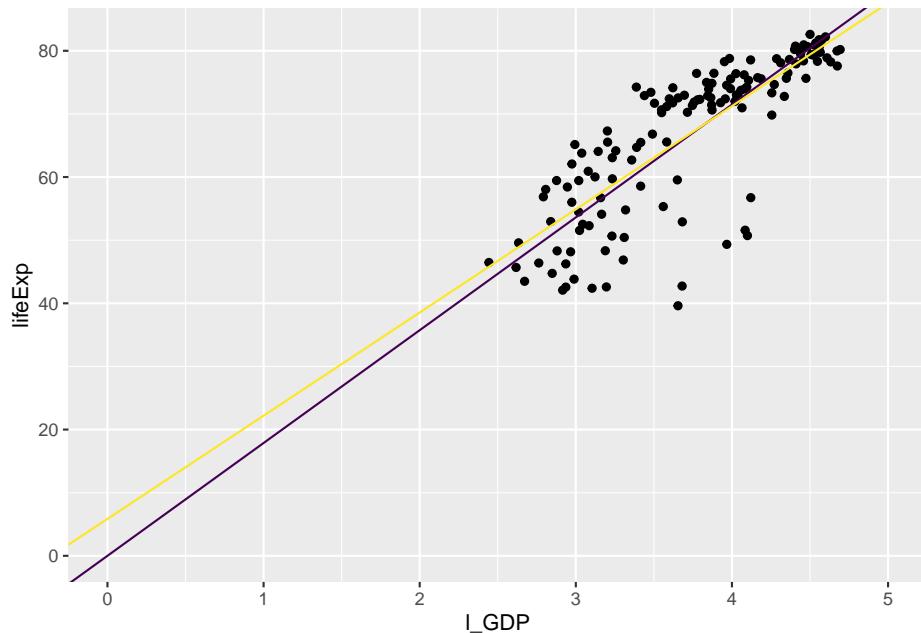


Figure 17.2: Log skaalas töötab nulli surutud interceptiga mudel sama hästi kui täismudel. See ei ole paraku mudeldamise üldine omadus.

```

b_gdp ~ dnorm(0, 10),
sigma ~ dcauchy(0, 2)
), data = g2007)

compare(gapmod4, gapmod5, gapmod6, gapmod7, gapmod8)
#>          WAIC pWAIC dWAIC weight    SE   dSE
#> gapmod7  965   3.0   0.0   0.53 25.11    NA
#> gapmod8  966   3.8   0.2   0.47 25.37  2.56
#> gapmod6 1028   2.6  62.4   0.00 14.07 18.21
#> gapmod4 1113   1.5 148.1   0.00 12.09 23.18
#> gapmod5 1486   0.8 521.1   0.00  7.29 26.82

```

Kuna Bayesi mudelite fittimine on keerulisem kui `lm()` abil, on eriti tähtis fittitud mudel välja plottida. See on esimene kaitselin lollide vigade ja halvasti jooksvate Markovi ahelate vastu.

Kui Bayesi mudeliteid on raskem fittida, siis milleks me peaksime neid eelistama tavalistele vähimruutude meetodil fittitud mudelitele? Tegelikult alati ei peagi. Aga siiski, Bayesi mudelid sisaldavad eksplitsiitset veakomponenti (σ), mis on kasulik mudelist uusi andmeid ennustades. Samuti annavad nad parima hinnangu ebakindlusele parameetrite väärustete hinnangute ümber, võimaldavad

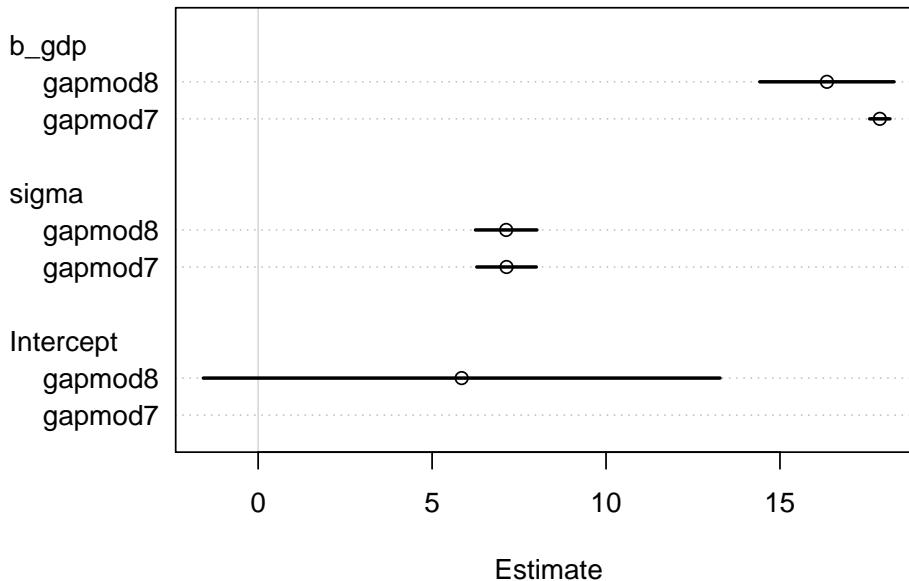


Figure 17.3: Mudelite võrdlusplot.

mudeli fittimisel siduda andmeid taustainfoga (prior) ning, mis kõige tähtsam, võimaldavad paindlikumalt fittida hierarhilisi mudeliteid (nende juurde tuleme hiljem).

Samas, kui prior on väheinformatiivne, siis Bayesi hinnangud mudeli koefitsientide kõige töenäolisematele väärustustele on praktiliselt samad, kui vähimruutude meetodiga `lm()` abil saadud punkt-hinnangud.

Siin me fitime pedagooglistel kaalutlustel kõike Bayesiga aga praktikas jätabavad paljud mõistlikud inimesed Bayesi hierarhiliste mudelite jaoks ja kasutavad lihtsate mudelite jaoks `lm()`.

Tagasi gapmod7 ja gapmod8 mudelite juurde. Plotime nende koefitsiendid koos usalduspiiridega.

```
plot(coeftab(gapmod7, gapmod8))
```

Pane tähele, et gapmod8 "b_gdp" koefitsiendi posteerior on palju laiem kui gapmod7 "b_gdp" oma. See on üldine nähtus, mis tuleneb sellest, et gapmod7-s on vähem parameetreid. **Iga lisatud parameeter kipub vähendama teiste parameetrite hindamise täpsust.**

Ennustused mudelist

Kuidas plottida meie hinnangud ebakindlusele parameetri tegeliku väärtsuse ümber? Siin tuleb appi `rethinking::link()`.

Nii tömbame posteriorist igale meie andmetes esinevale log GDP väärtsusele vastavad 1000 ennustust keskmise eluea kohta sellel `l_GDP` väärtsusel:

```
linked <- link(gapmod8)
linked <- as_tibble(linked)
linked_mean <- apply(linked, 2, HPDI, prob = 0.95)
```

Sel viisil saab tabeli, kus igale 142-le andmepunktist vastab üks veerg, milles on 1000 posteeriorist arvutatud ennustust `lifeExp` väärtsusele.

Praktikas soovime aga enamasti meie poolt ette antud `l_GDP` väärtsustel põhinevaid ennustusi keskmise eluea kohta. See käib nii:

```
# link() draws from the posterior 1000 mu values for each l_GDP value in the width obj
mu1 <- as_tibble(link(gapmod8, data = list(l_GDP = seq(2, 6, 0.1))))
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
#> Warning: `as_tibble.matrix()` requires a matrix with column names or a `.name_repair` argument.
#> This warning is displayed once per session.
```

Nüüd on meil `mu1` objektis 41 `l_GDP` väärust, millest igale vastab 1000 ennustust keskmise eluea kohta sellel `l_GDP`-l. Järgmiseks arvutame igale neist 41-st tulbast keskmise ja 95% HPDI ning plotime need koos andmepunktidega kasutades base-R graafikasüsteemi.

Pane tähele, et hall riba näitab ebakindlust ennustuse ümber keskmisele elueale üle kõikide riikide, mis võiksid sellist `l_GDP`-d omada (ehk ebakindlust regresioonijoonele). Kui me aga tahame ennustada ka keskmiste eluigade varieeruvust riigi tasemel (kasutades Bayesi hinnangut sigma parameetrile), siis on meil vaja `sim()` funktsiooni:

```
mu.mean <- apply(mu1, 2, mean) # applies the FUN mean() to each column
mu.HPDI <- apply(mu1, 2, HPDI, prob = 0.95) %>%
  t() %>%
  as_data_frame()
#> Warning: `as_data_frame()` is deprecated, use `as_tibble()` (but mind the new semantics)
```

```

#> This warning is displayed once per session.
mu.HPDI <- bind_cols(width = seq(2, 6, 0.1), mu.HPDI)
colnames(mu.HPDI) <- c("width", "lower", "upper")
sim.length <- as_tibble(sim(gapmod8, data = list(l_GDP = seq(2, 6, 0.1))))
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
height.PI <- sapply(sim.length, PI, prob = 0.95, simplify = FALSE) %>%
  do.call(rbind,. )
height.PI <- cbind(width = seq(2, 6, 0.1), height.PI) %>% as_tibble()
colnames(height.PI) <- c("width", "lower", "upper")

library(viridis)
ggplot(g2007) +
  geom_point(aes(l_GDP, lifeExp, color = continent)) +
  geom_line(data = data_frame(width = seq(2, 6, 0.1), mu.mean), aes(width, mu.mean)) +
  geom_ribbon(data = mu.HPDI, aes(x = width, ymin = lower, ymax = upper),
              fill = "grey5", alpha = 0.1) +
  geom_ribbon(data = height.PI, aes(x = width, ymin = lower, ymax = upper),
              fill = "grey50", alpha = 0.1) +
  labs(caption = "Dark grey, 95% HDPI - highest posterior density.\nLight grey, 95% PI - percenti",
       theme(legend.title = element_blank()) +
  scale_color_viridis(discrete = TRUE)
#> Warning: `data_frame()` is deprecated, use `tibble()``.
#> This warning is displayed once per session.

```

Nüüd ütleb laiem hall ala, et me oleme üsna kindlad, et nende riikide puhul, mille puhul mudel töötab, kohtame individuaalsete riikide keskmiseid eluigasid halli ala sees ja mitte sealts väljas. Nagu näha, on meil ka riike, mis jäädavad hallist alast kaugele ja mille keskmine eluiga on kövasti madalam, kui mudel ennustab. Need on äkki riigid, kus parasjagu on sõda üle käinud ja mille eluiga ei ole näiteks seetõttu SKP-ga lihtsas põhjuslikus seoses. Igal juhul tasuks need ükshaaval üle vaadata sest punktid, mida mudel ei seleta, võivad varjata endas mõnd huvitavat saladust, mis pikisilmi ootab avastajat. Lisaks: pane tähele, et mudel eeldab, et riikide keskmise eluea SD on muutumatu igal GDP väärtsuse.

Kuidas saada ennustusi kindlale l_GDP väärtsusele? Näiteks tulp V10 vastab l_GDP väärtsusele 2.9. Järgnevalt arvutame oodatavad keskmised eluead sellele SKP väärtsusele (fiksionaalsetele riikidele, millel võiks olla täpselt selline SKP):

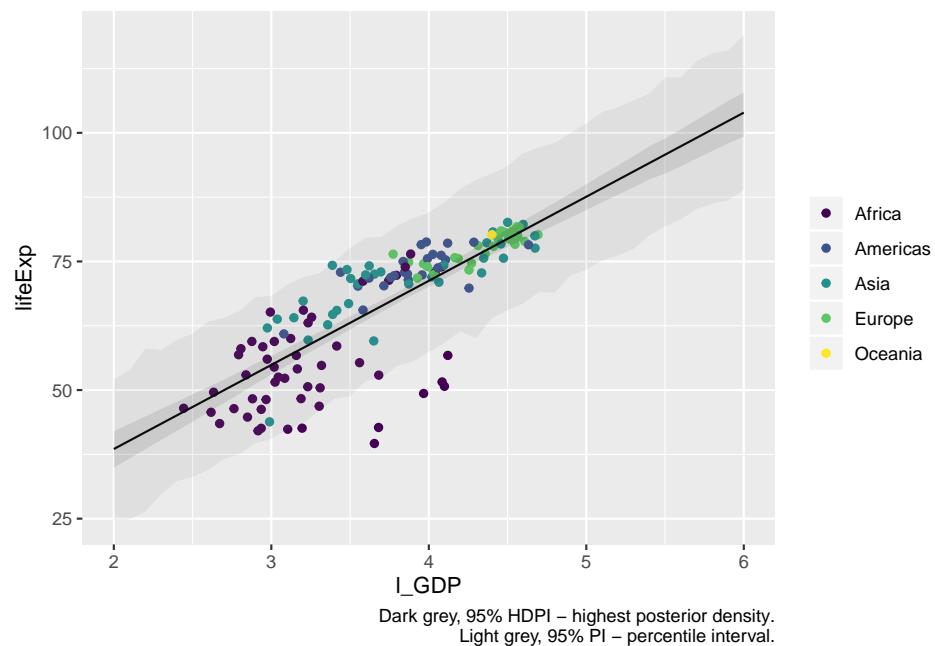


Figure 17.4: Ennustused mudelist.

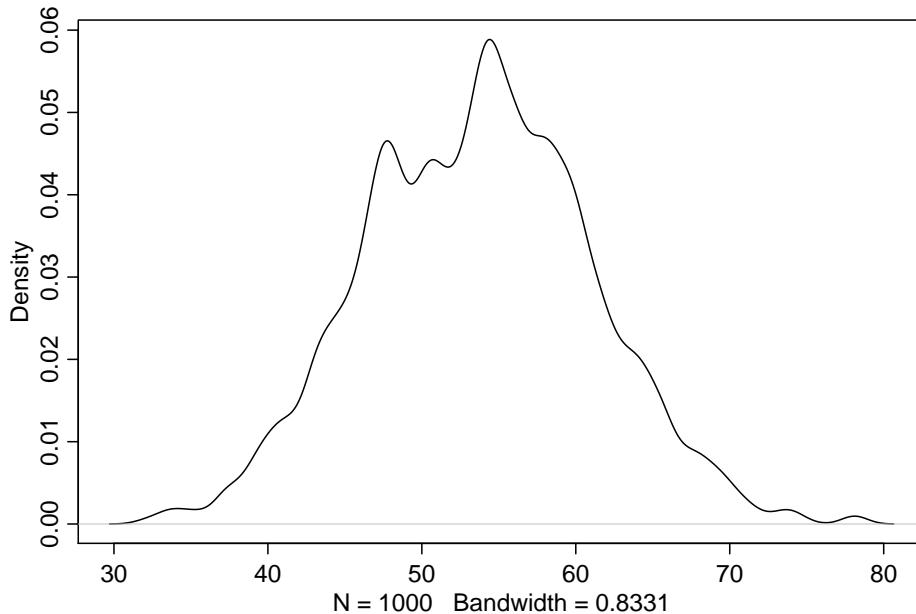


Figure 17.5: Ennustus mudelist kindlale log GDP väärustusele.

```

dens(sim.length$V10)

HPDI(sim.length$V10, prob = 0.95)
#> [1] 0.95 0.95
#> [2] 39.7 68.2

```

Nagu näha, võib mudeli kohaselt sellise riigi keskmise eluiga tulla nii madal, kui 40 aastat ja nii kõrge kui 67 aastat.

Lognormaalne töepäramudel

See mudel on alternatiiv andmete logaritmimisele, kui Y-muutuja (see muutuja, mille väärustust te ennustate) on lognormaalsete jaotusega.

Lognormaalne Y-i töepäramudel on mittelineaarse. Lognormaaljaotus defineeritakse üle mu ja sigma, mis aga vastavd hoopis $\log(Y)$ normaaljaotuse mu-le ja sigmale.

Lognormaalsete mudeli koefitsiendid mu ja sigma annavad keskmise ja standardhälbe $\log(y)$ -le, mitte y-le! Seega tuleb need koefitsiendid lineaarse mudeli andmeskaalas tõlgendamiseks ümber arvutada.

$$e^\mu = \mu^*$$

kus μ^* on geomeetriseline keskmise ehk mediaan

$$e^\sigma = \sigma^*$$

kus σ^* on multiplikatiivne standardhälve

μ^* korda/jagatud σ^* katab 68% lognormaaljaotusest, ja 2 σ^* katab 96% jaotusest.

Lognormaaljaotuse parameetrite põhjal on võimalik arvutada ka tavalised additiivsed keskväärtused ja standardhälbed, aga oluliselt keerulisemate valemitega:

$$e^\mu e^{\frac{\sigma^2}{2}} = \mu$$

kus saadud μ on tavaline additiivne keskväärtus (aritmeetiline keskmise)

$$e^{\sigma^2} (e^{\sigma^2} - 1) e^{2\sigma} = \sigma$$

ja saadud σ on tavaline additiivne standardhälve

Seekord ennustame GDP-d keskmise eluea põhjal (mis, nagu näha jooniselt, ei ole küll päris lognormaalne).

Mustaga on näidatud empiiriline SKP jaotus, rohelisega fititud lognormaalne mudel sellest samast jaotusest. Järgnevalt ennustame SKP-d keskmise eluea põhjal, milleks fitime lognormaalse töepäramudeli, kus mu on ümber defineeritud regressioonivõrrandiga:

```
m_ln1 <- map2stan(
  alist(
    gdpPercap ~ dlnorm( mu , sigma ),
    mu <- a + b * lifeExp,
    a ~ dnorm( 0, 10 ),
    b ~ dnorm( 0, 10 ),
    sigma ~ dcauchy( 0, 2 )
  ),
  data = g2007,
  start = list( a = 3, b = 0, sigma = 0.5 )
)

precis(m_ln1)
#>      Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> a     2.48   0.38     1.87     3.08   283    1
#> b     0.09   0.01     0.08     0.10   287    1
#> sigma 0.81   0.05     0.74     0.88   294    1
```

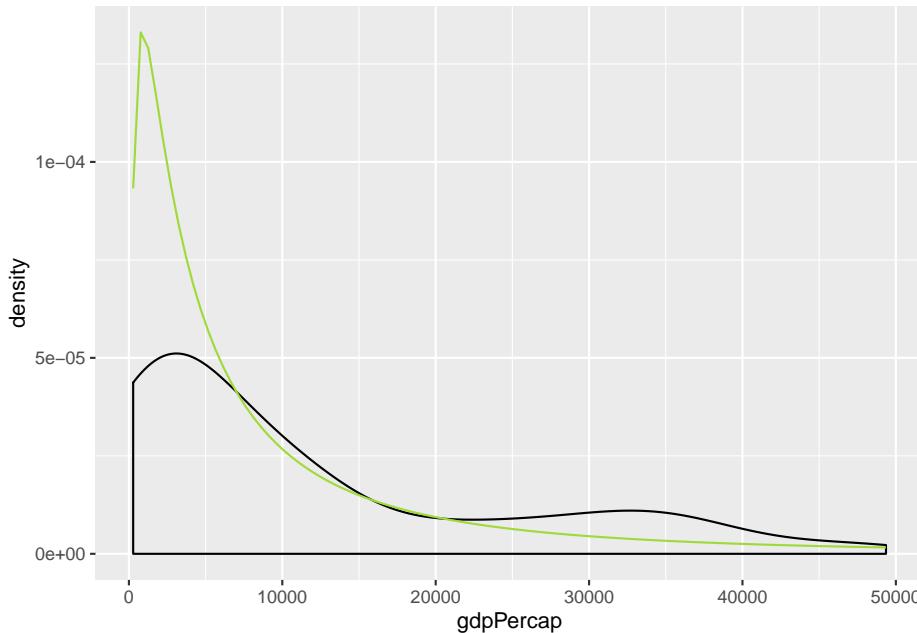


Figure 17.6: SKP-de jaotus

Kuna meil on tegemist mitte-lineaarse mudeliga, võltub tõusu (β) väärustus ka mudeli interceptist: $\beta = \exp(\alpha + \beta) - \exp(\alpha)$. See ei ole lineaarne seos: β omab seda suuremat mõju efektile (tõusule), mida suurem on α .

Kui meil on tegu binaarse prediktoriga, siis kodeerime selle 2 taset kui -1 ja 1. Sellises mudelis on tõus sama, mis efekti suurus ES, ja $ES = \exp(\alpha + \beta) - \exp(\alpha - \beta)$

```
a <- seq( 0, 10, length.out = 1000 )
b <- 2
b1 <- 3
y <- exp( a + b ) - exp( a )
y1 <- exp( a + b1 ) - exp( a )

plot( a, y, type = "l", xlab = "a value", ylab = "slope" )
lines( a, y1, col = "red" )
```

Must joon näitab mudeli tõusu sõltuvust parameetri a väärustusest, kui parameeter $b = 2$. Punane joon teeb sedasama, kui $b = 3$.

Selline on siis mudeli tõusude (beta) posteerior:

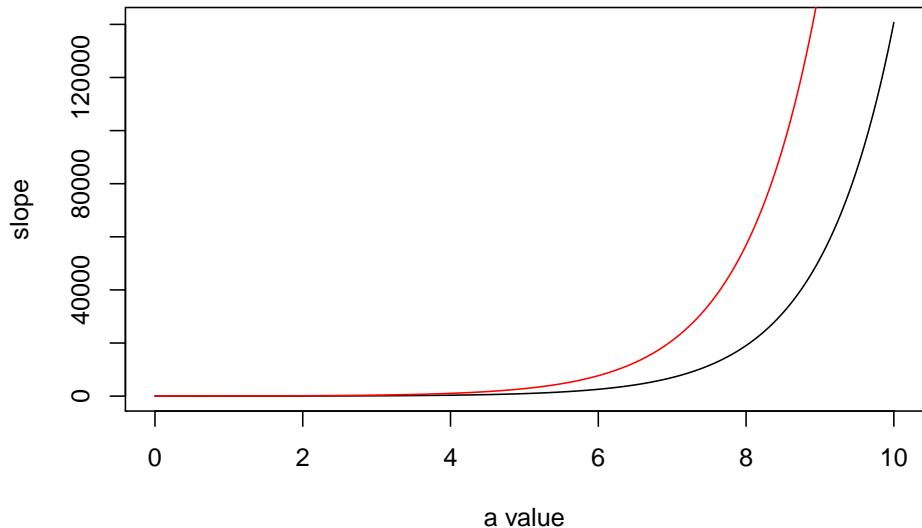


Figure 17.7: Mudeli tõus sõltub interceptist.

```
s_ln1 <- extract.samples(m_ln1) %>% as.data.frame()
beta <- exp(s_ln1$a + s_ln1$b) - exp(s_ln1$a)

dens(beta)
```

Lognormaaljaotusega mudelis täidab normaaljaotusega mudeli intercepti rolli eelkõige mediaan, mis on defineeritud kui $\exp(a)$, aga arvutada saab ka keskmise:

```
i_median <- exp(s_ln1$a)
mean(i_median)
#> [1] 12.8
i_mean <- exp(s_ln1$a + (s_ln1$sigma ^ 2) / 2)
mean(i_mean)
#> [1] 17.8
```

Sin ennustame fititud mudelist uusi andmeid (väljamõeldud riikide rikkust):

```
sim_ci <- sim(m_ln1) %>%
  as_tibble() %>%
  apply(2, HPDI, prob = 0.95)
#> [1] 100 / 1000
[200 / 1000]
[300 / 1000]
[400 / 1000]
[500 / 1000]
[600 / 1000]
```

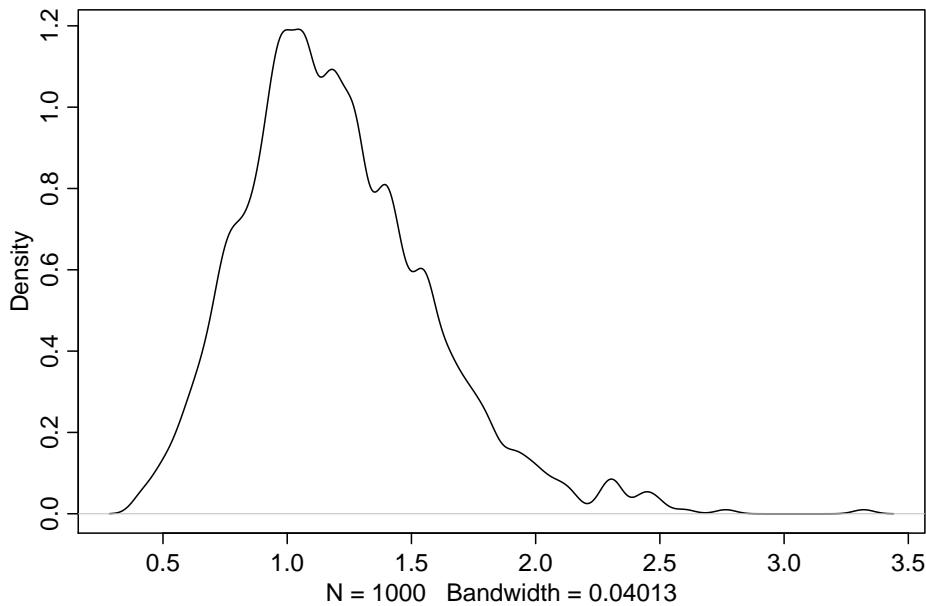


Figure 17.8: Mudeli tõusude (beta) postseerior.

```
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
ggplot(g2007, aes(lifeExp, gdpPercap)) +
  geom_point(aes(color = continent), size = 0.8) +
  geom_ribbon(aes( ymin = sim_ci[1,], ymax = sim_ci[2,]), alpha = 0.2) +
  scale_color_viridis(discrete = TRUE)
```

Ka see mudel jäab hänta Aafrika outlieritega, mille eluiga ei suuda ennustada rikkust.

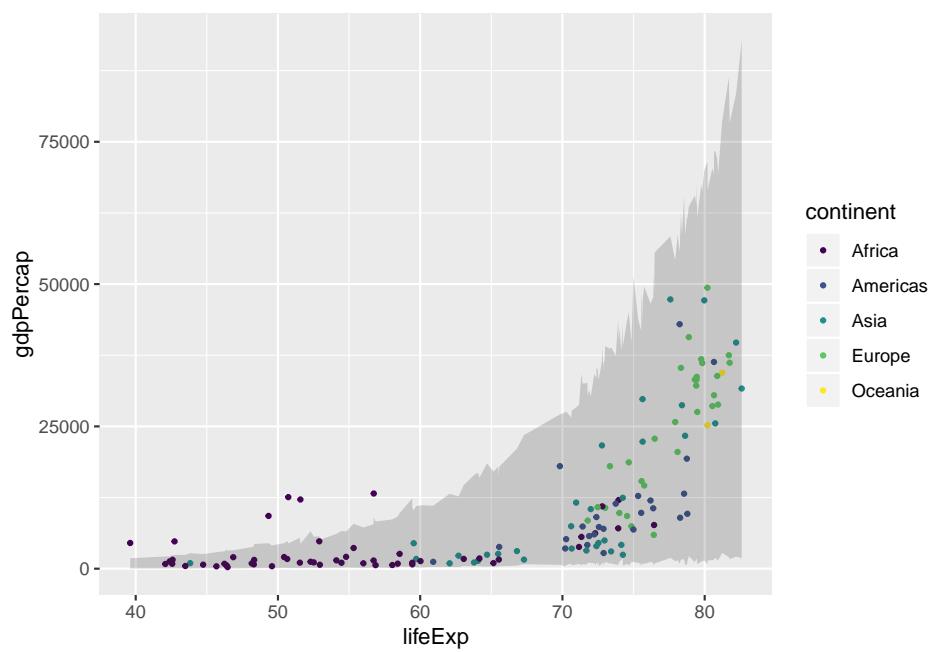


Figure 17.9: Ennustus mudelist.

Chapter 18

Mitme prediktoriga lineaarne regressioon

```
library(tidyverse)
library(gapminder)
library(rethinking)
library(BayesVarSel)
library(mice)
library(leaps)
```

Vaatame jälle gapminderi andmeid aastast 2007. Meil on võimalik lisada regressioonivõrrandisse lisaprediktoreid. Nüüd ei küsi me enam, kuidas mõjutab l_GDP varieeruvus keskmise eluea varieeruvust vaid: kuidas mõjutavad muutujad l_GDP, continent ja logaritm pop-ist (rahvaarvust) keskmist eluiga. Me modelleerime selle lineaarselt nii, et eeldusena varieeruvad need x-i muutujad üksteisest sõltumatult: $y = a + b_1x_1 + b_2x_2 + b_3x_3$

```
g2007 <- gapminder %>%
  filter(year == 2007)
g2007 <- g2007 %>%
  mutate(l_GDP = log10(gdpPerCap),
        l_pop = log10(pop),
        lpop_s = (l_pop - mean(l_pop)) / sd(l_pop),
        lGDP_s = (l_GDP - mean(l_GDP)) / sd(l_GDP)) %>%
  as.data.frame()
```

Sellise mudeli tõlgendus on suhteliselt lihtne:

koef b1 ütleb meile, kui mitme ühiku võrra tõuseb/langeb muutuja y (eluiga) kui muutuja x1 (l_GDP) tõuseb 1 ühiku võrra; tingimusest, et me hoiame kõigi teiste muutujate väärtsused konstantsed.

208CHAPTER 18. MITME PREDIKTORIGA LINEAARNE REGRESSIOON

Sarnane definitsioon kehtib ka kõigi teiste prediktorite (x-de) kohta.

Kui meil on mudelis SKP ja pop (rahvaarv), siis saame küsida

- 1) kui me juba teame SKP-d, millist ennustuslikku lisaväärtust annab meile ka populatsiooni suuruse teadmine? ja
- 2) kui me juba teame populatsiooni suurust, millist lisaväärtust annab meile ka SKP teadmine?

Järgenval mudelil on 4 parameetrit (intercept + 3 betat).

```
m1 <- lm(lifeExp ~ l_GDP + continent + l_pop, data = g2007)
summary(m1)
#>
#> Call:
#> lm(formula = lifeExp ~ l_GDP + continent + l_pop, data = g2007)
#>
#> Residuals:
#>    Min      1Q  Median      3Q     Max
#> -19.425 -2.246 -0.014  2.468 14.957
#>
#> Coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 19.4182    7.4557   2.60   0.0102 *
#> l_GDP        10.6876   1.2378   8.63  1.5e-14 ***
#> continentAmericas 11.6564   1.6929   6.89  2.0e-10 ***
#> continentAsia    10.0521   1.5776   6.37  2.7e-09 ***
#> continentEurope   11.2320   1.9265   5.83  3.9e-08 ***
#> continentOceania  12.8918   4.5493   2.83   0.0053 **
#> l_pop          0.0928   0.8076   0.11   0.9087
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 5.95 on 135 degrees of freedom
#> Multiple R-squared:  0.767, Adjusted R-squared:  0.757
#> F-statistic: 74.2 on 6 and 135 DF, p-value: <2e-16
```

loeme mudelis “+” märki nagu “või”. Ehk, “eluiga võib olla funktsioon SKP-st **või** rahvaarvust”.

Intercept 19 ei tähenda tõlgenduslikult midagi. l-GDP tõus ühiku võrra tõstab eluiga 10.7 aasta võrra.

Võrdluseks lihtne mudel

```
m2 <- lm(lifeExp ~ l_GDP, data = g2007)
summary(m2)
#>
#> Call:
```

```
#> lm(formula = lifeExp ~ l_GDP, data = g2007)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -25.95  -2.66   1.22   4.47 13.12
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 4.95      3.86   1.28    0.2
#> l_GDP       16.59     1.02  16.28 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 7.12 on 140 degrees of freedom
#> Multiple R-squared:  0.654, Adjusted R-squared:  0.652
#> F-statistic: 265 on 1 and 140 DF, p-value: <2e-16
```

Siin on l_GDP mõju suurem, 16.6 aastat. Millisel mudelil on siis õigus? Proovime veel ülejäänud variandid

```
m3 <- lm(lifeExp ~ l_GDP + continent, data = g2007)
summary( m3 )
#>
#> Call:
#> lm(formula = lifeExp ~ l_GDP + continent, data = g2007)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -19.492 -2.315 -0.043  2.550 14.882
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 20.14      4.03   4.99 1.8e-06 ***
#> l_GDP        10.66     1.21   8.78 6.1e-15 ***
#> continentAmericas 11.69     1.65   7.07 7.5e-11 ***
#> continentAsia    10.11     1.48   6.85 2.3e-10 ***
#> continentEurope   11.27     1.89   5.95 2.1e-08 ***
#> continentOceania  12.93     4.52   2.86  0.0049 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 5.93 on 136 degrees of freedom
#> Multiple R-squared:  0.767, Adjusted R-squared:  0.759
#> F-statistic: 89.7 on 5 and 136 DF, p-value: <2e-16

m4 <- lm(lifeExp ~ l_GDP + l_pop, data = g2007)
```

```
AIC(m1, m2, m3, m4)
#>      df AIC
#> m1   8 918
#> m2   3 965
#> m3   7 916
#> m4   4 962
```

Võitja mudel on hoopis m3, mis võtab arvesse kontinendi. Siin on l_GDP mõju samuti 10.7 aastat. Lisaks näeme, et kui riik ei asu Aafrikas, siis on l_GDP mõju elueale u 11 aasta võrra suurem. Seega elu Aafrika kisub alla keskmise eluea riigi rikkusest sõltumata. Võib olla on põhjuseks sõjad, võib-olla AIDS ja malaaria, võib-olla midagi muud.

Millise mudeli me peaksime siis avaldama? Vastus on, et need kõik on olulised, et vastata küsimusele, millised faktorid kontrollivad keskmist eluiga? Mudelite võrdlusest näeme, et rahvaarvu mõju elueale on väike või olematu ning et SKP mõju avaldub log skaalas (viitab teatud tüüpi eksponenttsiaalsele protsessidele, kus rikkus tekitab uut rikkust) ning, et Aafrikaga on midagi pahasti ja teistmoodi kui teiste kontinentidega. Aafrikast tasub otsida midagi, mida meie senised mudelid ei kajasta.

Miks ei ole mudeli summary tabelis Aafrikat? Põhjus on tehniline. Kategoorilisi muutujaid, nagu kontinent, vaatab mudel paariviisilises võrdluses, mis tähendab et k erineva tasemega muutujast tekitatakse $k - 1$ uut muutujat, millest igaühel on kaks taset (0 ja 1). See algne muutuja, mis üle jäääb (antud juhul Africa), jäääb ilma oma uue muutujata. Me saame teisi uusi kontinendi põhjal tehtud muutujaid tölgendada selle järgi, kui palju nad erinevad Africa-st.

Miks multivariaatsed mudelid head on?

- 1) nad aitavad kontrollida nn kaasnevaid (*confounding*) muutujaid. Kaasnev muutuja võib olla korreleeritud mõne teise muutujaga, mis meile huvi pakub. See võib nii maskeerida signaali, kui tekitada vööts-signaali, kuni y ja x_1 seose suuna muutmiseni välja.
- 2) ühel tagajärvel võib olla mitu põhjust.
- 3) Isegi kui muutujad ei ole omavahel üldse korreleeritud, võib ühe tähtsus sõltuda teise väärustest. Näiteks taimed vajavad nii valgust kui vett. Aga kui ühte ei ole, siis pole ka teisel suurt tähtsust.

Mudeldamine standardiseeritud andmetega

Kui me lahutame igast andmepunktist selle muutuja keskväärtuse siis saame 0-le tsentreeritud andmed. Kui me sellisel viisil saadud väärtsused omakorda läbi jagame muutuja standardhälbgaga, siis saame standardiseeritud andmed, mille keskväärtus on null ja SD = 1.

$$\text{Standard.andmed} = (x - \text{mean}(x)) / \text{sd}(x)$$

Nii on lihtsam erinevas skaalas muutujaid omavahel võrrelda (1 ühikuline muutus võrdub alati muutusega 1 standardhäve võrra) ja mudeli arvutamine üle mcmc ahelate on ka lihtsam.

```
m5 <- map2stan(
  alist(
    lifeExp ~ dnorm( mu , sigma ) ,
    mu <- a + b_GDP * lGDP_s + b_pop * lpop_s ,
    a ~ dnorm( 0 , 10 ) ,
    c(b_GDP, b_pop) ~ dnorm( 0 , 1 ) ,
    sigma ~ dunif( 0 , 10 )
  ), data = g2007 )

precis(m5)
#>      Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> a     66.74   0.64    65.77    67.79  1000    1
#> b_GDP 6.95   0.58    6.06     7.84   775    1
#> b_pop  0.80   0.55   -0.11     1.58   870    1
#> sigma  7.64   0.50    6.82     8.40   876    1
```

kui l_GDP kasvab 1 sd võrra, siis eluiga kasvab 6.9 aasta võrra.

```
f1 <- glimmer(lifeExp ~ lGDP_s + lpop_s + continent, data = g2007)
#> alist(
#>   lifeExp ~ dnorm( mu , sigma ),
#>   mu <- Intercept +
#>     b_lGDP_s*lGDP_s +
#>     b_lpop_s*lpop_s +
#>     b_continentAmericas*continentAmericas +
#>     b_continentAsia*continentAsia +
#>     b_continentEurope*continentEurope +
#>     b_continentOceania*continentOceania,
#>   Intercept ~ dnorm(0,10),
#>   b_lGDP_s ~ dnorm(0,10),
#>   b_lpop_s ~ dnorm(0,10),
#>   b_continentAmericas ~ dnorm(0,10),
#>   b_continentAsia ~ dnorm(0,10),
#>   b_continentEurope ~ dnorm(0,10),
#>   b_continentOceania ~ dnorm(0,10),
```

212CHAPTER 18. MITME PREDIKTORIGA LINEAARNE REGRESSIOON

```
#>      sigma ~ dcauchy(0,2)
#> )
```

See on mudeli struktuur, mis sisaldab uusi kategoorilisi muutujaid

Sin on tähtis anda map2stan()-le ette glimmeri poolt eeltöödeldud andmed:

```
m6 <- map2stan(
  f1$f,
  data = f1$d)

precis(m6)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> Intercept      59.92   0.97   58.31   61.42   344    1
#> b_LGDP_s        6.29   0.68    5.20    7.31   450    1
#> b_lpop_s        0.06   0.50   -0.75    0.79  1000    1
#> b_continentAmericas 11.66   1.53    9.07   13.94   436    1
#> b_continentAsia    10.11   1.47    7.87   12.51   428    1
#> b_continentEurope   11.26   1.83    8.50   14.26   434    1
#> b_continentOceania  11.18   4.00    4.87   17.50   784    1
#> sigma             5.95   0.36    5.38    6.49   931    1
```

18.1 Prediktorite valik e milline on parim mudel

Kui me võtame oma mudelisse prediktoreid, mis ei aita märkimisväärselt ennustada y-muutuja väärtsusi, siis lisame sellega mudelisse müra ja meie tulemus saab sellest ainult kannatada. Seega tasub enne formaalset mudelit välja visata mõttetud muutujad. Seda püüame nüüd teha kahel meetodil.

Kasutame diabeedi andmeid, kus 403 USA Lõunaosariikide neegril mõõdeti 15 muutujat, mis seostuvad ülekaalu, diabeedi ja teiste kardiovaskulaarsete riskifaktoritega.

Willem JP, Saunders JT, DE Hunt, JB Schorling: Prevalence of coronary heart disease risk factors among rural blacks: A community-based study. Southern Medical Journal 90:814-820; 1997

Schorling JB, Roach J, Siegel M, Baturka N, Hunt DE, Guterbock TM, Stewart HL: A trial of church-based smoking cessation interventions for rural African Americans. Preventive Medicine 26:92-101; 1997.

Kõigepealt bayesi meetodil. Eeldame tavalist mitme prediktoriga iseseisvat (aditiivset) lineaarset regressiooni (ilma interaktsioonideta v mitmetasemeliste mudeliteta).

```
library(BayesVarSel)
library(mice)
```

```

diabetes <- read.csv2("data/diabetes.csv")
imp_d <- mice(diabetes, m = 1, print = FALSE)
diab <- complete(imp_d)
diab_Bvs <- Bvs(formula = hdl ~ chol + stab.glu + ratio + glyhb + age + gender + height + weight
#> Info. . .
#> Most complex model has 16 covariates
#> From those 1 is fixed and we should select from the remaining 15
#> chol, stab.glu, ratio, glyhb, age, gendermale, height, weight, framelarge, framemedium, framesmall
#> The problem has a total of 32768 competing models
#> Of these, the 10 most probable (a posteriori) are kept
#> Working on the problem...please wait.

library(car)
vif(lm(hdl ~ chol + stab.glu + ratio + glyhb + age + height + weight + bp.1s + bp.1d + waist, data = diab))
#>          chol      stab.glu      ratio      glyhb      age      height      weight      bp.1s
#>        1.42       2.32       1.51       2.44       1.64       1.20       4.87       2.13
#>         bp.1d      waist
#>        1.77       4.69

summary(diab_Bvs)
#>
#> Call:
#> Bvs(formula = hdl ~ chol + stab.glu + ratio + glyhb + age + gender +
#>       height + weight + frame + bp.1s + bp.1d + waist + hip, data = diab)
#>
#> Inclusion Probabilities:
#>           Incl.prob. HPM MPM
#> chol              1   *   *
#> stab.glu          0.0243
#> ratio             1   *   *
#> glyhb             0.0191
#> age               0.0384
#> gendermale        0.0165
#> height            0.0396
#> weight            0.0745
#> framelarge        0.0366
#> framemedium       0.0136
#> framesmall         0.0218
#> bp.1s              0.015
#> bp.1d              0.0136
#> waist              0.0484
#> hip                0.4649
#> ---
#> Code: HPM stands for Highest posterior Probability Model and
#> MPM for Median Probability Model.

```

214 CHAPTER 18. MITME PREDIKTORIGA LINEAARNE REGRESSIOON

```
#>

Aga mis juhtub, kui me eemaldame muutuja chol?
diab_Bvs2 <- Bvs(formula = hdl ~ stab.glu + ratio + glyhb + age + gender + height + we
#> Info. . .
#> Most complex model has 15 covariates
#> From those 1 is fixed and we should select from the remaining 14
#> stab.glu, ratio, glyhb, age, gendermale, height, weight, framelarge, framemedium, f
#> The problem has a total of 16384 competing models
#> Of these, the 10 most probable (a posteriori) are kept
#> Working on the problem...please wait.
summary(diab_Bvs2)
#>
#> Call:
#> Bvs(formula = hdl ~ stab.glu + ratio + glyhb + age + gender +
#>       height + weight + frame + bp.1s + bp.1d + waist + hip, data = diab)
#>
#> Inclusion Probabilities:
#>           Incl.prob. HPM MPM
#> stab.glu          0.0645
#> ratio              1   *   *
#> glyhb             0.124
#> age                0.9775   *   *
#> gendermale        0.1301
#> height             0.0663
#> weight             0.1769
#> framelarge         0.697   *   *
#> framemedium       0.1246
#> framesmall         0.0858
#> bp.1s              0.0943
#> bp.1d              0.6342   *   *
#> waist               0.1331
#> hip                 0.174
#> ---
#> Code: HPM stands for Highest posterior Probability Model and
#> MPM for Median Probability Model.
#>
```

Nüüd ilmusid välja lisamuutujad, mis äkki on olulised ja selgu, et vanus on peaaegu sama oluline kui ratio!

Järgneb **Hüpoteeside testimine**, mis annab Bayesi faktorid (B) nullhüpoteesi suhtes. B=1 tähendab, et kaks hüpoteesi on andmete poolt võrdselt toetatud. B = 100 tähendab, et see hüpotees, mis ei ole H₀, on andmete poolt 100 korda rohkem toetatud, kui H₀. B = 0.1 tähendab, et hüpotees on 10 korda vähem toetatud kui H₀. Bayesi faktor mõõdab tõendusmaterjali (evidence), mis on

suhteline mõõt, millega andmed toetavad ühte hüpoteesi rohkem või vähem kui teist hüpoteesi.

H_0 on ilma prediktoriteta mudel, mis annab tulemuseks keskmise hdl -i.

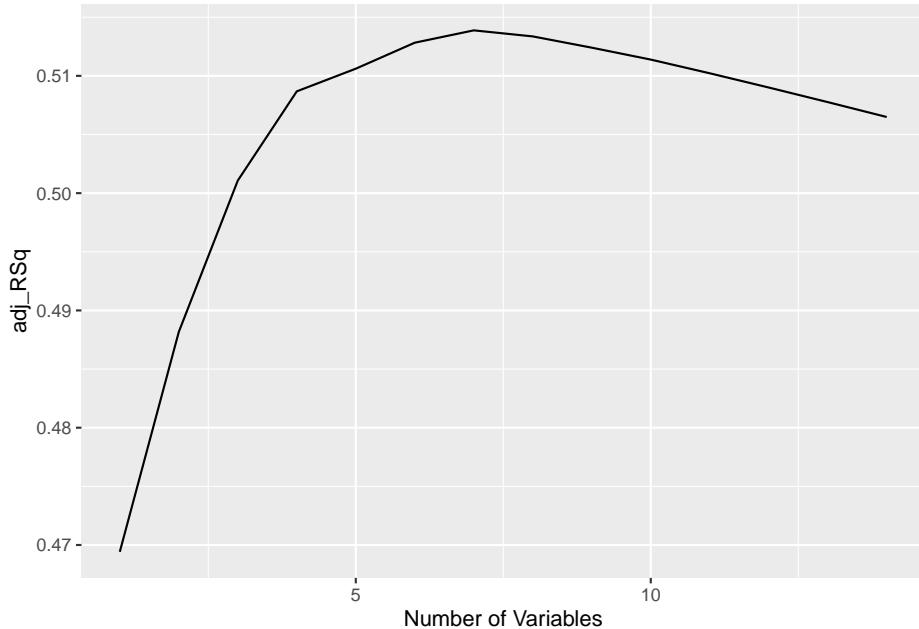
```
fullmodel <- hdl ~ chol + stab.glu + ratio + glyhb + age + gender + height + weight + frame + bp.1s + bp.1d + waist + hip
reducedmodel <- hdl ~ chol + ratio
reducedmodel2 <- hdl ~ age + ratio + frame
nullmodel <- hdl ~ 1
Btest(models = c(H0 = nullmodel, H1 = fullmodel, H2 = reducedmodel, H3 = reducedmodel2), data = data)
#> -----
#> Models:
#> $H0
#> hdl ~ 1
#>
#> $H1
#> hdl ~ chol + stab.glu + ratio + glyhb + age + gender + height +
#>      weight + frame + bp.1s + bp.1d + waist + hip
#>
#> $H2
#> hdl ~ chol + ratio
#>
#> $H3
#> hdl ~ age + ratio + frame
#>
#> -----
#> Bayes factors (expressed in relation to H0)
#> H0.to.H0 H1.to.H0 H2.to.H0 H3.to.H0
#> 1.00e+00 9.23e+130 1.38e+142 1.41e+55
#> -----
#> Posterior probabilities:
#> H0 H1 H2 H3
#> 0 0 1 0
```

Väga huvitav, $chol$ muutuja eemaldamine (millest ma kahtlustan, et see on osaliselt(?) redundantne $ratio$ muutujaga) langetas Bayesi faktorit meeletult.

sageduslik alternatiiv muutujate valimisele:

```
library(leaps)
regfit.full<- regsubsets(hdl ~ stab.glu + ratio + glyhb + age + gender + height + weight + frame
                           + bp.1s + bp.1d + waist + hip, nbopt=100)
reg.summary<-summary(regfit.full)
regs <- data.frame(adj_RSq = reg.summary$adjr2)
ggplot(regs) + geom_line(aes(x=1:nrow(regs), y = adj_RSq)) + xlab("Number of Variables")
```

216 CHAPTER 18. MITME PREDIKTORIGA LINEAARNE REGRESSIOON



Peale 2. muutuja lisamist jäab adjusteeritud r-ruut stabiilseks. Seega piisab kahest muutujast.

```
reg.summary$adjr2
#> [1] 0.469 0.488 0.501 0.509 0.511 0.513 0.514 0.513 0.512 0.511 0.510
#> [12] 0.509 0.508 0.506

summary(regfit.full)
#> Subset selection object
#> Call: regsubsets.formula(hdl ~ stab.glu + ratio + glyhb + age + gender +
#> height + weight + frame + bp.1s + bp.1d + waist + hip, data = diab,
#> numax = 19, method = "backward")
#> 14 Variables (and intercept)
#>          Forced in Forced out
#> stab.glu      FALSE      FALSE
#> ratio        FALSE      FALSE
#> glyhb        FALSE      FALSE
#> age          FALSE      FALSE
#> gendermale   FALSE      FALSE
#> height        FALSE      FALSE
#> weight        FALSE      FALSE
#> framelarge    FALSE      FALSE
#> framemedium  FALSE      FALSE
#> framesmall    FALSE      FALSE
#> bp.1s         FALSE      FALSE
#> bp.1d         FALSE      FALSE
```

```

#> waist      FALSE    FALSE
#> hip       FALSE    FALSE
#> 1 subsets of each size up to 14
#> Selection Algorithm: backward
#>          stab.glu ratio glyhb age gender male height weight frame large
#> 1  ( 1 ) " "   "*"  " "   " "   " "   " "   " "   " "
#> 2  ( 1 ) " "   "*"  " "   "*"  " "   " "   " "   " "
#> 3  ( 1 ) " "   "*"  " "   "*"  " "   " "   " "   "*" 
#> 4  ( 1 ) " "   "*"  " "   "*"  " "   " "   " "   "*" 
#> 5  ( 1 ) " "   "*"  " "   "*"  " "   " "   " "   "*" 
#> 6  ( 1 ) " "   "*"  " "   "*"  "*"  " "   " "   "*" 
#> 7  ( 1 ) " "   "*"  " "   "*"  "*"  " "   " "   "*" 
#> 8  ( 1 ) "*"   "*"  " "   "*"  "*"  " "   " "   "*" 
#> 9  ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  " "   "*" 
#> 10 ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  " "   "*" 
#> 11 ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  " "   "*" 
#> 12 ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  "*"  " " 
#> 13 ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  "*"  "*" 
#> 14 ( 1 ) "*"   "*"  " "   "*"  "*"  " "   "*"  "*"  "*" 

#>          framemedium framesmall bp.1s bp.1d waist hip
#> 1  ( 1 ) " "   " "   " "   " "   " "   " "
#> 2  ( 1 ) " "   " "   " "   " "   " "   " "
#> 3  ( 1 ) " "   " "   " "   " "   " "   " "
#> 4  ( 1 ) " "   " "   " "   "*"  " "   " "   " "
#> 5  ( 1 ) " "   " "   " "   "*"  " "   " "   "*" 
#> 6  ( 1 ) " "   " "   " "   "*"  " "   " "   "*" 
#> 7  ( 1 ) " "   " "   " "   "*"  " "   " "   "*" 
#> 8  ( 1 ) " "   " "   " "   "*"  " "   " "   "*" 
#> 9  ( 1 ) " "   " "   " "   "*"  " "   " "   "*" 
#> 10 ( 1 ) "*"   " "   " "   "*"  " "   " "   "*" 
#> 11 ( 1 ) "*"   " "   " "   "*"  " "   " "   "*" 
#> 12 ( 1 ) "*"   " "   " "   "*"  " "   " "   "*" 
#> 13 ( 1 ) "*"   "*"   " "   "*"  " "   " "   "*" 
#> 14 ( 1 ) "*"   "*"   " "   "*"  " "   "*"  " "

```


Chapter 19

Keerulisemate mudelitega töötamine

```
library(tidyverse)
library(gapminder)
library(rethinking)
library(bayesplot)
library(gridExtra)
```

Kasuta graafilisi meetodeid. Mudeli koefitsientide jäollitamine üks ei päästa.

```
g2007 <- gapminder %>%
  filter(year == 2007)
g2007 <- g2007 %>%
  mutate(l_GDP = log10(gdpPercap),
        l_pop = log10(pop),
        lpop_s = (l_pop - mean(l_pop)) / sd(l_pop),
        lGDP_s = (l_GDP - mean(l_GDP)) / sd(l_GDP)) %>%
  as.data.frame()
```

Predictor residual plots

```
m5 <- map2stan(
  alist(
    lifeExp ~ dnorm( mu , sigma ) ,
    mu <- a + b_GDP * lGDP_s + b_pop * lpop_s ,
    a ~ dnorm( 0 , 10 ) ,
    c(b_GDP, b_pop) ~ dnorm( 0 , 1 ) ,
```

```
    sigma ~ dunif( 0 , 10 )
), data = g2007 )
```

Plotime varieeruvuse, mida mudel ei oota ega seleta.

```
names(coef(m5))
#> [1] "a"      "b_GDP"  "b_pop"  "sigma"
```

Kõigepealt lihtne residuaalide plot, kus meil on y-teljel residuaalid ja x-teljel X1 muutuja tegelikud valimiväärtused. $Y = 0$ tähistab horisontaalse joonena mudeli ennustatud Y (eluea) väärtsusi kõigil prediktori X1 (lGDP_s) väärustel ja residuaal on defineeritud kui tegelik Y miinus mudeli poolt ennustatud eluiga sellel X1 väärusel. Mudeli ennustuse saamiseks anname mudelile ette fikseeritud parameetrite (koefitsientide) a, b_GDP ja b_pop väärtsused ning arvutame oodataava keskmise eluea üle kõigi valimis leiduvate lGDP_s ja lpop_s väärustele. Seega saame sama palju keskmise eluea ennustusi, kui palju on meie andmetabelis ridu.

```
# Using the fitted model compute the expected value of y (mu)
# for each of the 142 data rows.
mu <- coef(m5)[ 'a' ] +
  coef(m5)[ 'b_GDP' ] * g2007$lGDP_s +
  coef(m5)[ 'b_pop' ] * g2007$pop_s

# compute residuals - a vector w. 142 values
m.resid <- g2007$lifeExp - mu

ggplot( g2007, aes( lGDP_s, m.resid ) ) +
  geom_segment( aes( xend = lGDP_s, yend = 0 ), size = 0.2 ) +
  geom_point( size = 0.5, type = 1 )
#> Warning: Ignoring unknown parameters: type
```

Me näeme, et seal kus SKP on väiksem kipuvad residuaalid olema negatiivsed, mis tähendab, et mudel ülehindab keskmist eluiga. Ja vastupidi, seal kus SKP on üle keskmise, mudel kipub alahindma keskmist eluiga.

See osos tuleb eriti selgelt välja järgmisel pildil, kus plotime residuaalide sõltuvuse elueast (kui eelmise plot oli m.resid ~ X1, siis nüüd plotime m.resid ~ Y). Lisaks joonistame selguse mõttes regressioonisirge. Kui residuaalid oleks ühtlaselt jaotunud mõlemale poole mudeli ennustust, siis saaksime horisontaalse regressioonisirge. Tegeliku sirge töüs näitab, et suuremad eluead omavad eelistatult poikiivseid residuaale ja väiksemad eluead negatiivseid residuaale. See tähendab, et mudel alahindab eluiga seal, kus SKP on kõrge ja vastupidi, ülehindab eluiga seal, kus SKP on madal.

```
g2007$m.resid <- m.resid
ggplot(g2007, aes(lifeExp, m.resid)) +
  geom_smooth(method = "lm", se = FALSE) +
```

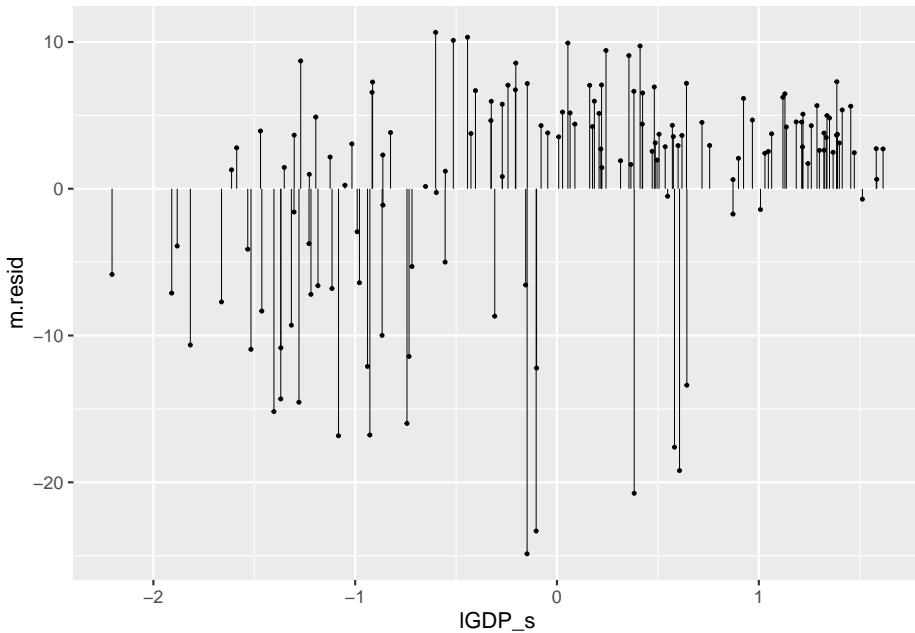


Figure 19.1: Mudeli residuaalide plot ($m.\text{resid} \sim X1$).

```
geom_point() +
  geom_hline(yintercept = 0, color = "grey", linetype = 2)
```

Horisontaalne punktiirjoon näitab, kus mudel vastab täpselt andmetele.

Ennustavad plotid

Plot, kus me ennustame keskmise eluea sõltuvust SKP-st nii riikide kaupa eraldi (andmepunktide paupa) kui üldiselt kõikide riikide keskmisena, millel on mingi kindel SKP (mudeli parima ennustuse ehk sirge asendi ümber valitsevat ebakindlust). Et seda teha, hoiame rahvaarvu konstantsena oma keskväärtusel, mis standardiseeritud andmetl võrdub alati nulliga. `link()` funktsioon annab meile keskmiste eluigade ennustused meie poolt ette antud X1 ja X2 väärustel, ning `sim()` annab meile eluigade ennustused fiktiosaalsete riikide kaupa samadel X1 ja X2 väärustel. Nagu näha, on meie mudeli arvates riikide kaupa ennustamine palju laiemal varieeruvusega kui üle kõikväimalike riikide kesmise kaupa ennustamine.

```
# prepare new counterfactual data
pred.data <- tibble(
```

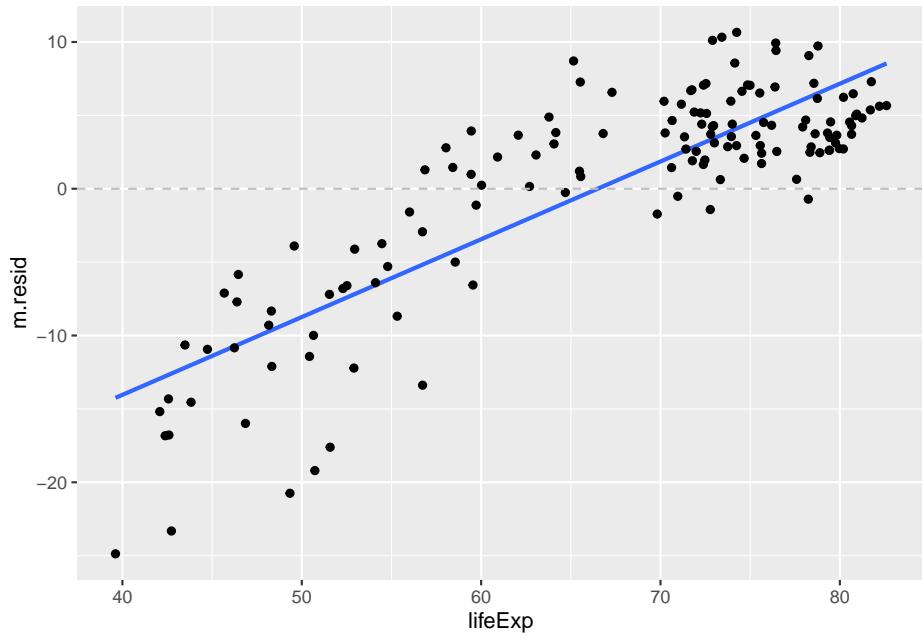


Figure 19.2: m.resid ~ Y plot

```

lgdp_s = seq(-3, 3, length.out = 30), # need meie poolt valitud lgdp_s väärustused,
lpop_s = 0 # rahvaaru fikseeritakse muutuja keskmisele tasemele, mis standardiseer
)

# compute counterfactual mean lifeExp (mu)
mu <- link(m5, data = pred.data)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
mu.mean <- apply(mu, 2, mean)
mu.PI <- apply(mu, 2, PI)

# simulate counterfactual lifeExpectancies of individual countries
R.sim <- sim(m5, data = pred.data)

```

```
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
R.sim <- na.omit(R.sim)
R.PI <- apply(R.sim, 2, PI)
pred.data$mu.mean <- mu.mean
pred.data$lower <- mu.PI[1,]
pred.data$upper <- mu.PI[2,]
pred.data$lower1 <- R.PI[1,]
pred.data$upper1 <- R.PI[2,]

ggplot(pred.data, aes(lGDP_s, mu.mean)) +
  geom_line() +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "grey60", alpha = 0.3) +
  geom_ribbon(aes(ymin = lower1, ymax = upper1), fill = "grey10", alpha = 0.3)
```

Näeme, kuidas ennustus sobib/ei sobi andmetega. Võrdle eelneva ennustuspildiga, kus mudel ei sisalda rahvaarvu. Ennustuse intervallid on originaalandmete skaalas (aastates), mis on hea.

Posterior prediction plots

Posterioorsed ennustusplotid panevad kõrvuti (või üksteise otsa) Y-i algandmed ja mudeli ennustused Y-väärtustele. Kui meie valimi suurus on N, siis me tõmbame mudelist näiteks 5 valimit, igaüks suurusega N ja plotime need kõrvuti valimiandmete plotiga. Siis me vaatame sellele plotile peale ja otsustame, kas mudeli ennustused on piisavalt läheidal valimi andmetele. Kui ei, siis on tõenäoline, et meie mudelis on midagi mäda ja me peame hakkama sealt vigu otsima. Tösi küll, keerulisemate hierarhiliste mudelite korral on vahest raske otsustada, millised peaksid tulema eduka mudeli ennustused võrreldes algandmetega — aga siiski, see on arvatavasti kõige tähtsam plot, mida oma mudelist teha!

- 1) võrdle mudeli ennustusi andmetega. (Aga arvesta sellega, et mitte kõik mudelid ei püüagi täpselt andmetele vastata.)

```
yrep <- sim(m5)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
```

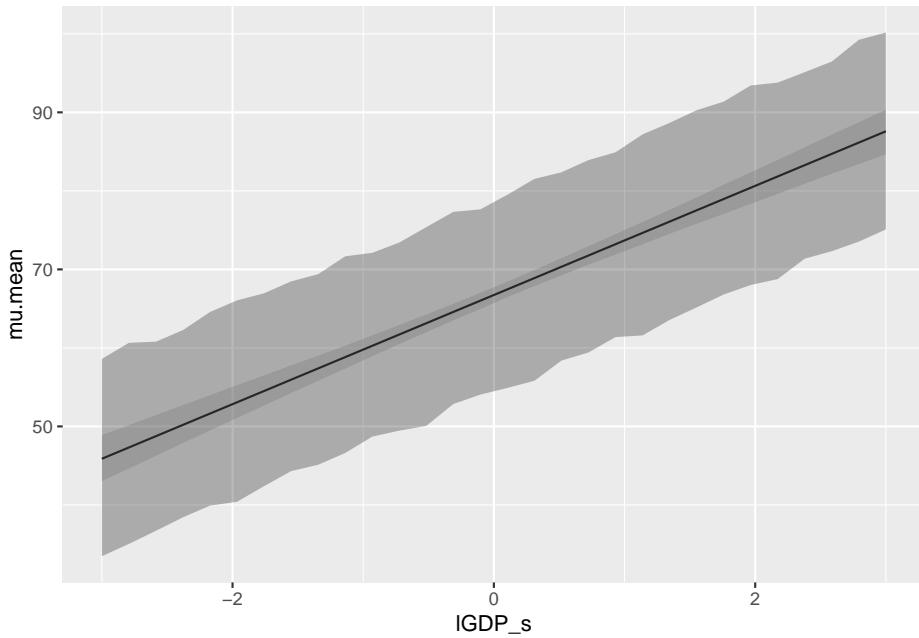


Figure 19.3: Ennustav plot

```
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
ppc_dens(g2007$lifeExp, yrep[1:5, ])
```

- 2) Millisel viisil täpselt meie mudel ebaõnnestub? See plot annab mõtteid, kuidas mudelite parandada.

Ploti ennustused andmepunktide vastu, pluss jooned, mis näitavad igale ennustusele omistatud usaldusintervalli. Lisaks veel sirge, mis näitab täiuslikku ennustust (slope = 1, intercept = 0).

Ja nüüd plotime ennustused Y-le tegelike Y valimi väärustete vastu:

```
mu <- link(m5)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
```

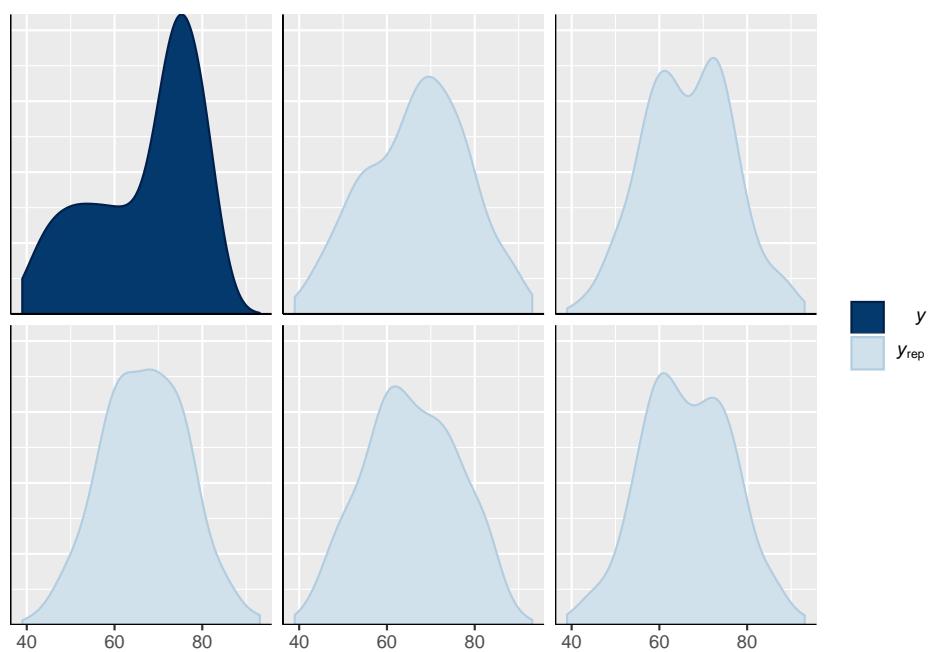


Figure 19.4: Valimi andmed vs. mudeli poolt ennustatud andmed.

```
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
mu.mean <- apply(mu, 2, mean)

mu.PI <- apply(mu , 2 , PI)

g2007$mu.mean <- mu.mean

ggplot(g2007, aes(lifeExp, mu.mean)) +
  geom_point() +
  geom_crossbar(ymin = mu.PI[1,], ymax = mu.PI[2,]) +
  geom_abline(intercept = 0, slope = 1, lty = 2) +
  ylab("Predicted life expectancy") +
  xlab("Observed life expectancy") +
  coord_cartesian( xlim=c( 40, 85 ), ylim=c( 40, 85 ))
```

Sin on ennustus ja seda ümbritsev ebakindlus iga riigi keskmisele elueale.

Järgnev plot annab ennustusvea igale riigile. Siin tähistab 89% CI näiteks Vietnamile eluigade vahemikku, millese jäab mudeli ennustuse kohaselt 89% kõikvõimalike fiktionsaalsete riikide keskmistest eluigadest, mille SKP ja rahvaarv võrdub Vietnami omaga. Kuna me tsentreerime CI Vietnami tegeliku keskmise eluea residuaalile (erinevusele mudeli ennustusest), näitab see, kui palju erineb Vietnami eluiga mudeli ennustusest riikidele, nagu Vietnam. See plot annab meile riigid, mille suhtes mudel jääni jäab. Enamasti leiame need riigid Aafrikast.

```
# compute residuals
life.resid <- g2007$lifeExp - mu.mean

mu_sim <- sim( m5 )
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

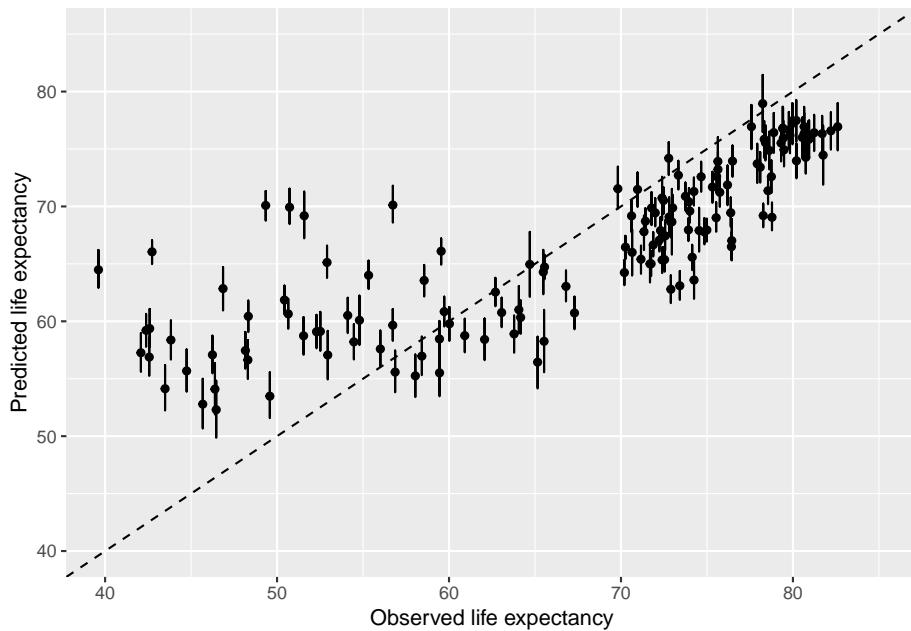


Figure 19.5: Ennustus vs. valimi vääritus

```

sim.PI <- apply( mu_sim , 2 , PI )

ggplot(g2007, aes(x = life.resid, y = reorder(country, life.resid))) +
  geom_point() +
  geom_errorbarh(aes(xmin = lifeExp - sim.PI[1,],
                      xmax = lifeExp - sim.PI[2,] ),
                 color = "red") +
  geom_vline(xintercept = 0) +
  theme(text = element_text(size = 7),
        axis.title.y = element_blank())

```

punased jooned näitavad 89% ennustuspiire igale residuaalile riigi tasemel (89% kõikvõimalike riikide keskmiste eluigade residuaalidest sellel SKPl jäab punasesse vahemikku).

Interaktsioonid prediktorite vahel

Eelnevad mudelid eeldavad, et prediktorite varieeruvused on üksteisest sõltumatud. Aga mis siis, kui see nii ei ole ja ühe prediktori mõju suurus sõltub

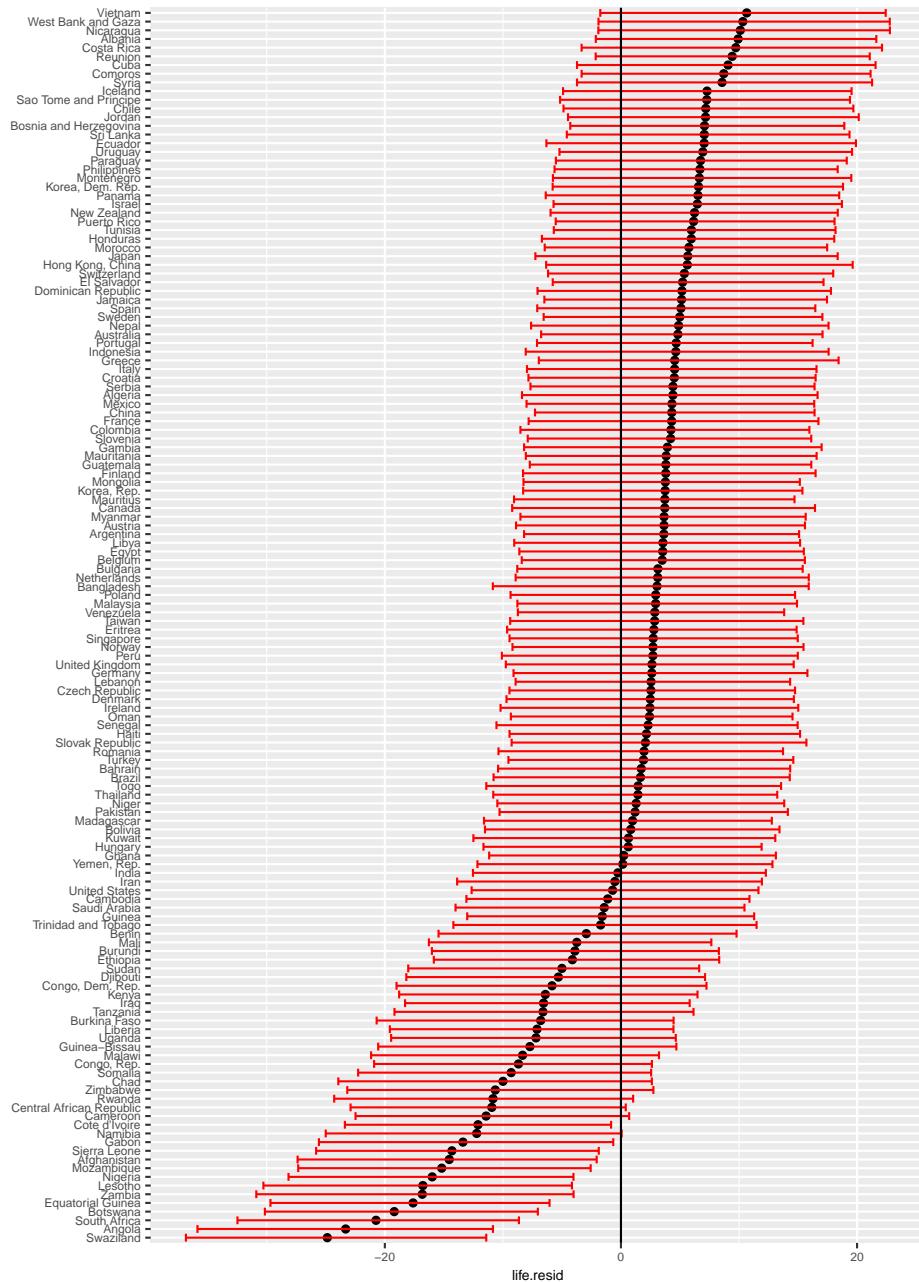


Figure 19.6: Ennustused riigi kaupa.

teisest prediktorist, ehk prediktorite vahel on interaktsioon? Lihtsaim viis sel-list interaktsiooni modelleerida on lisades interaktsiooni aditiivsele mudelile korrutamisetehtena:

$$y = a + b_1x_1 + b_2x_2 + b_3x_1x_2$$

Sellise mudeli järgi erineb sirge tõus b₁ erinevatel b₂ väärustel, ja erinevuse määär sõltub b₃-st (b₃ annab interaktsiooni tugevuse). Samamoodi ja sümmeetriliselt erineb ka tõus b₂ sõltuvalt b₁ väärustest. See on ühine paljude hierarhiliste mudelitega, mida võib omakorda vaadelda massivsete interaktsionimudelite na. Seevastu $y = a + b_1x_1 + b_2x_2$ tüüpi mudel annab b₁-le konstantse tõusunurga, kuid laseb intercepti muutuma sõltuvalt b₂ väärustest (ja vastupidi).

Interaktsionimudeli fittimises pole midagi erilist võrreldes sellega, mida me oleme juba õppinud. Aga fititud parameetrite tõlgendamine on keeruline. Alustame diskreetse muutujaga, continent, ja mudeldame selle interaktsiooni SKP-ga.

```
f1 <- glimmer(lifeExp ~ lgdp_s * continent, data = g2007)
#> alist(
#>   lifeExp ~ dnorm( mu , sigma ),
#>   mu <- Intercept +
#>     b_lgdp_s*lgdp_s +
#>     b_continentAmericas*continentAmericas +
#>     b_continentAsia*continentAsia +
#>     b_continentEurope*continentEurope +
#>     b_continentOceania*continentOceania +
#>     b_lgdp_s_X_continentsAmericas*lgdp_s_X_continentsAmericas +
#>     b_lgdp_s_X_continentsAsia*lgdp_s_X_continentsAsia +
#>     b_lgdp_s_X_continentsEurope*lgdp_s_X_continentsEurope +
#>     b_lgdp_s_X_continentsOceania*lgdp_s_X_continentsOceania,
#>   Intercept ~ dnorm(0,10),
#>   b_lgdp_s ~ dnorm(0,10),
#>   b_continentsAmericas ~ dnorm(0,10),
#>   b_continentsAsia ~ dnorm(0,10),
#>   b_continentsEurope ~ dnorm(0,10),
#>   b_continentsOceania ~ dnorm(0,10),
#>   b_lgdp_s_X_continentsAmericas ~ dnorm(0,10),
#>   b_lgdp_s_X_continentsAsia ~ dnorm(0,10),
#>   b_lgdp_s_X_continentsEurope ~ dnorm(0,10),
#>   b_lgdp_s_X_continentsOceania ~ dnorm(0,10),
#>   sigma ~ dcauchy(0,2)
#> )
m1 <- map2stan(f1$f, f1$d)
plot(precis(m1))
```

Aafrika on siin võrdluseks.

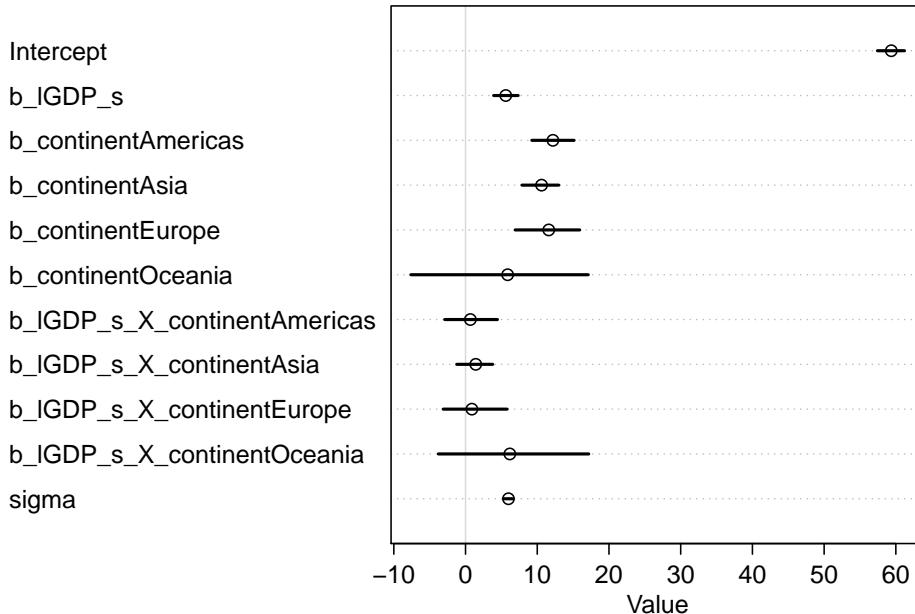


Figure 19.7: Mudeli koefitsientide plot.

Interaktsioon on sümmeetrisiline. Me võime sama hästi küsida, kui palju SKP mõju elueale sõltub kontinendist, kui seda, kui palju kontinendi mõju eluale sõltub SKP-st.

Nüüd joonistame välja regressioonisirge Aafrika ja Euroopa jaoks eraldi m1 mudeli põhjal

```
c1 <- coef(m1)
names(c1)
#> [1] "Intercept"                  "b_lGDP_s"
#> [3] "b_continentAmericas"      "b_continentAsia"
#> [5] "b_continentEurope"        "b_continentOceania"
#> [7] "b_lGDP_s_X_continentsAmericas" "b_lGDP_s_X_continentsAsia"
#> [9] "b_lGDP_s_X_continentsEurope"   "b_lGDP_s_X_continentsOceania"
#> [11] "sigma"
```

Kõigepealt defineerime X1 ja X2 väärtsused, millele teeme ennustused link() funktsiooni abil. Link tabelist veergude keskmise annab keskmise eluea ennustuse vastavale mandrile ja SKP-le. PI() abil saame 89% CI igale ennustusele.

```
dd <- as.data.frame(f1$d) #we use the dataframe made by glimmer()
#in dd all continents are in separate 2-level columns (except Africa)
dd1 <- dd %>% filter(continentAmericas == 0,
```

```

continentAsia == 0,
continentEurope == 0,
continentOceania == 0)
mu.Africa <- link(m1, dd1)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
mu.Africa.mean <- apply(mu.Africa, 2, mean)
mu.Africa.PI <- apply(mu.Africa, 2, PI, prob = 0.9)

ggplot(dd1, aes(lGDP_s, lifeExp)) +
  geom_point() +
  geom_ribbon(aes(ymin = mu.Africa.PI[1,], ymax = mu.Africa.PI[2,]), alpha = 0.15) +
  geom_line(aes(y = mu.Africa.mean))

dd1 <- dd %>% filter(continentEurope == 1)
mu.Europe <- link(m1, dd1)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
mu.Europe.mean <- apply( mu.Europe , 2 , mean )
mu.Europe.PI <- apply( mu.Europe , 2 , PI , prob=0.9 )

ggplot(data=dd1, aes(lGDP_s, lifeExp)) +
  geom_point()+
  geom_ribbon( aes(ymin=mu.Europe.PI[1,], ymax=mu.Europe.PI[2,]), alpha=0.15)+ 
  geom_line( aes( y=mu.Europe.mean))

```

Nagu näha, on meil nüüd üsna erinevad sirge tõusunurgad.

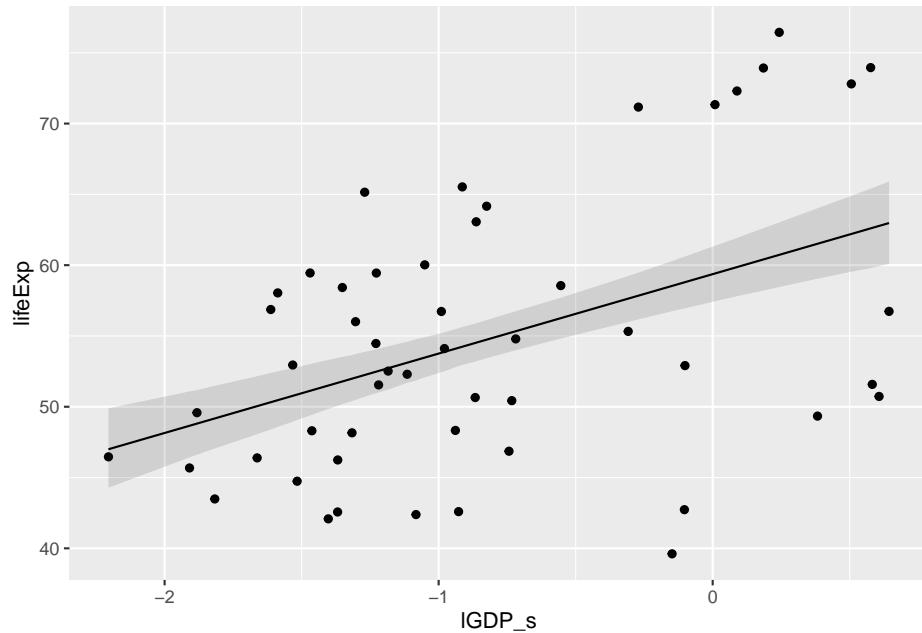


Figure 19.8: Ennustusplot Aafrikale.

Interaktsioonid pidevatele tunnustele

Kasutame standardiseeritud prediktoreid, sest nende koefitsiente saab paremini tõlgendada (teglikult piisab prediktorite tsentreerimisest). Meie andmed käsitlevad diabeedimarkereid Ameerika lõunaosariikide neegritel 1960-ndatel. Me ennustame siin sõltuvalt vanusest ja vööümbermõõdust hdl-i — high density cholesterol — mis on nn hea kolesterol.

```
diabetes <- read.csv2("data/diabetes.csv")
d1 <- diabetes %>% select(hdl, age, waist) %>% na.omit()
d2 <- d1 %>% mutate(age_st = (age - mean(age)) / sd(age),
                      waist_st = (waist - mean(waist)) / sd(waist))

m2 <- map2stan(
  alist(
    hdl ~ dnorm( mu , sigma ) ,
    mu <- a + bR*age_st + bA*waist_st + bAR*age_st*waist_st,
    a ~ dnorm( 0, 100 ),
    bR ~ dnorm( 0, 2 ),
    bA ~ dnorm( 0, 2 ),
    bAR ~ dnorm( 0, 2 ),
    sigma ~ dcauchy( 0, 1 )
```

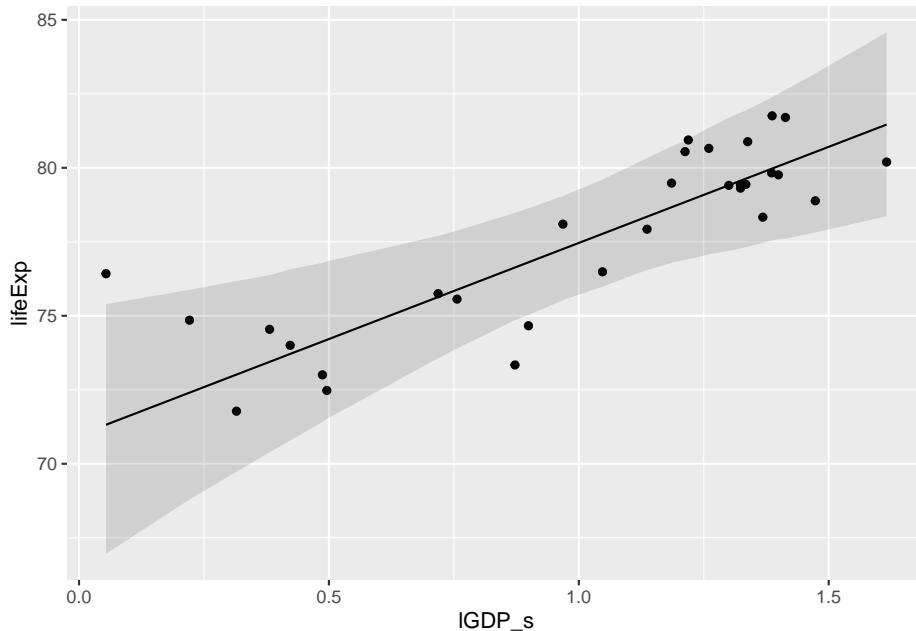


Figure 19.9: Ennustusplot Euroopale.

```
) , data = d2)
plot(precis(m2))
```

NB! Järgmised interpretatsioonid kehtivad ainult siis, kui mudeldame nullile tsentreeritud andmeid.

a - hdl-i oodatav keskväärtus siis kui võõ-übermanööt ja vanus on fikseeritud oma keskmistel väärustel. bR - oodatav hdl-i muutus, kui vanus kasvab 1 aasta vörra ja võõ-übermanööt on fikseeritud oma keskväärtusel bA - sama, kui võõ-übermanööt kasvab 1 ühiku (inch) vörra bAR - kaks ekvivalentset tõlgendust: 1) oodatav muutus vanuse mõju määrale hdl-le, kui võõ-übermanööt kasvab 1 ühiku vörra. 2) oodatav muutus võõ-übermanöödu mõju määrale hdl-le, kui vanus kasvab 1 ühiku vörra.

Negatiivne bAR tähendab, et vanus ja võõ-übermanööt omavad vastandlikke mõjusid hdl-i tasemele, aga samas kumgki tõstab teise tähtsust hdl-le.

```
m3 <- map2stan(
  alist(
    hdl ~ dnorm(mu, sigma),
    mu <- a + bR * age_st + bA * waist_st,
    a ~ dnorm(0, 100),
```

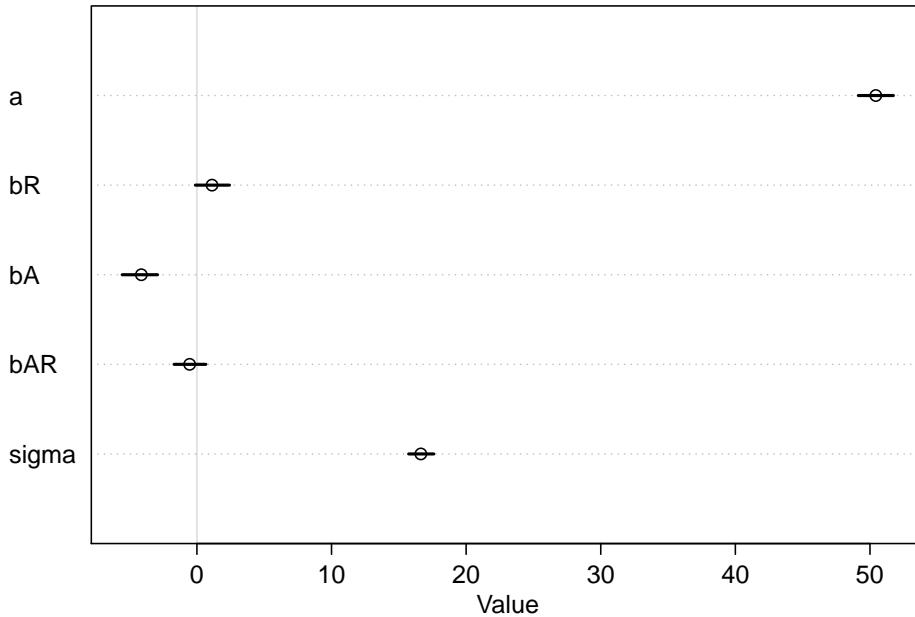


Figure 19.10: mudeli koefitsientide plot

```

c(bR, bA) ~ dnorm(0, 2),
sigma ~ dcauchy(0, 1)
), data = d2)

compare(m2, m3)
#>   WAIC pWAIC dWAIC weight    SE   dSE
#> m3 3391    5.6    0.0    0.72 41.8    NA
#> m2 3393    7.1    1.9    0.28 41.9 1.91

```

Siin on tegelikult eelistatud ilma interaktsioonita mudel. Aga kuna interaktsioonimudeli kaal on ikkagi 28%, tasub meil ennustuste tegemisel mõlemat mudelit koos arvestada vastavalt oma kaalule.

```

coeftab(m2, m3)
#>      m2      m3
#> a      50.4    50.4
#> bR     1.12   1.09
#> bA    -4.13  -4.10
#> bAR    -0.54    NA
#> sigma   16.6   16.6
#> nobs     400    400

```

Tõesti, bA ja bR on mõlemas mudelis väga sarnased. m3 on kindlasti lihtsamini tõlgendatav.

Ensemble teeb ära nii link()-i kui sim()-i, kasutades mõlemat mudelit vastavalt nende mudelite WAIC-i kaaludele ja toodab listi, mille elementideks on link() toodetud maatriks ja sim() toodetud maatriks.

Teeme 3 plotti: waist = 0 (keskmine), waist = -1 (miinus üks sd) ja waist = 1

```
waist_fun <- function(waist, ...) {
  d.pred <- data.frame(age_st = seq( -2, 2, length.out = 20 ),
                        waist_st = waist)
  e <- ensemble(..., data = d.pred)
  hdl <- apply(e$link, 2, mean)
  mu.PI <- apply(e$link, 2, PI, prob = 0.97)
  ggplot(d.pred, aes(x = age_st)) +
    geom_line(aes(y = hdl)) +
    geom_line( aes(y = mu.PI[1,]), linetype = 2) +
    geom_line( aes( y = mu.PI[2,] ), linetype = 2) +
    ylim(40, 70)
}
```

Ensemble mudel:

```
## Fit ensemble model
# p <- lapply(-1:1, waist_fun, m2, m3)
## Plot three plots
# do.call(grid.arrange, c(p, ncol = 3))
##
p_1 <- waist_fun(-1, m2, m3)
p0 <- waist_fun(0, m2, m3)
p1 <- waist_fun(1, m2, m3)
grid.arrange(p_1, p0, p1, ncol = 3)
#> Warning: Removed 2 rows containing missing values (geom_path).
```

Ja sama ainult ühe mudeliga – m2.

```
w0 <- waist_fun(-1, m2)
w_1 <- waist_fun(0, m2)
w1 <- waist_fun(1, m2)
grid.arrange(w0, w_1, w1, ncol = 3)
```

Nüüd on hästi näha, et interaktsionimudel laseb sirge tõusunurgad vabaks!

Üldiselt tasub interaktsioon mudelisse sisse kirjutada siis, kui see interaktsioon on teoreetiliselt mõtekas (ühe prediktori mõju võiks sõltuda teise prediktori tasemest). Interaktsiooni koefitsiendi määramine võib suurendada ebakindlust teiste parameetrite määramisel, seda eriti siis kui interaktsiooni parameeter on korreleeritud oma komponentide parameetritega (vt pairs(model)).

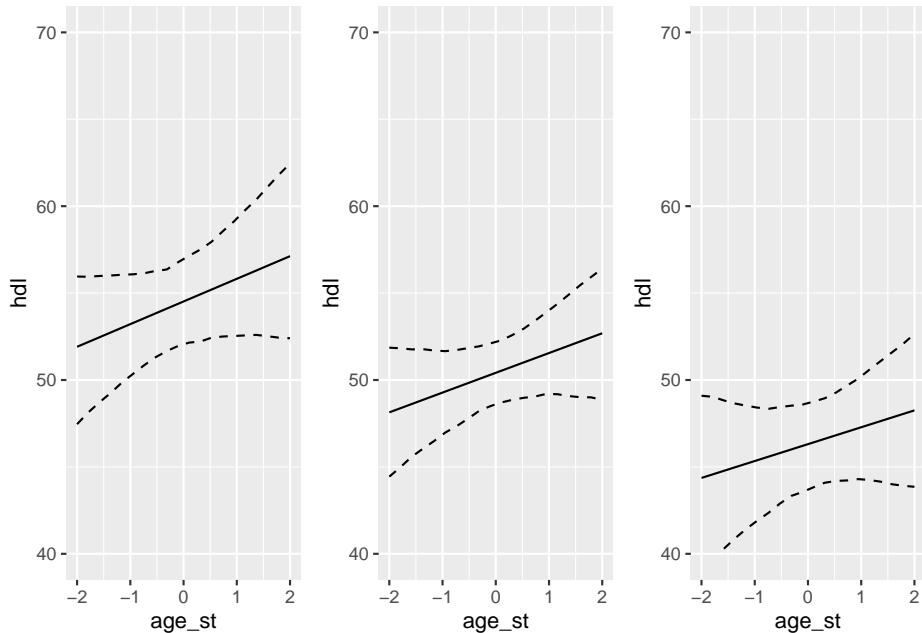


Figure 19.11: Ennustusplot üle kahe mudeli.

Isegi kui interaktsiooniparameetri posteerior hõlmab 0-i, tuleb interaktsiooni parameetrit mudelisse pannes arvestada, et individuaalsete prediktorite mõju ei saa summeerida pelgalt läbi nende koefitsientide. Selle asemel tuleb vaadata sirge töusu erinevatel teiste prediktorite väärustustel (nagu eelneval joonisel)

Kui tavalline interaktsioonimudel on $y = a + b_1x_1 + b_2x_2 + b_3x_1x_2$, siis mis juhtub, kui meie mudel on $y = b_1x_1 + b_3x_1x_2$? See tähdab, et me surume b_2 väärustuse nulli, mis võib ära rikkuda mudeli teiste parameetrite posteeriorid! Kui teil on alust arvata, et b_2 -l puudub otsene mõju y väärustusele (kuid tal on mõju b_1 väärustusele), siis võib muidugi ka sellist mudelit kasutada. Aga see on haruldane juhtum.

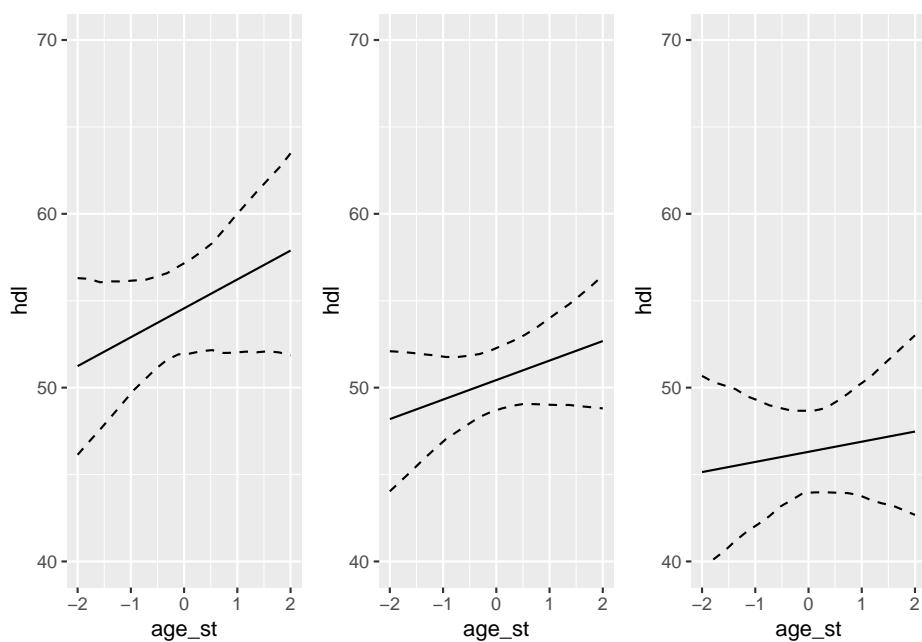


Figure 19.12: Ennustusplot m2-le.

Chapter 20

Mitmetasemelised mudelid

Mitmetasemeline mudel on regressioonimudel, kus andmed on struktureeritud gruppidesse ja mudeli koefitsiendid võivad erineda grupist gruppi.

Statistika teoria ütleb, et me peaksime oma mudelitesse hõlmama need faktorid, mida kasutati eksperimendi disainis. Mitmetasandilised mudelid on parim viis, kuidas mudelisse panna katsedisainis esinevaid erinevatel tasemetel klastreid ja samas võltaida mudeli ülefittimist. Mitmetasandiline mudel kajastab sellise katse või vaatluse struktuuri, kus andmed ei grupeeru mitte ainult katse- ja kontrolltingimuste vahel, vaid ka lisaklastritesse ehk gruppidesse. Näiteks, kui me mõõdame platseebo-kontrollitud uuringus kümmet patsienti ja teeme igale patsiendile viis kordusmõõtmist (kahetasemeline mudel). Või kui meil on geeniuring, kus uuritakse korraga 1000 valgu taset ja uuring toimub 10 laboris (3-tasemeline mudel). Või kui mõõdame kalamaksaõli mõju matemaatikaeksami tulemustele kümnes koolis, ja igas neist viies klassis (3-tasemeline mudel).

Tavapärane lähenemine oleks kõigepealt keskmistada andmed iga klassi sees ning seejärel keskmistada iga kooli sees (võtta igale koolile 5 klassi keskmise). Ning seejärel, võttes iga kooli keskmise üheks andmepunktiks, teha soovitud statistiline test ($N = 10$, sest meil on 10 kooli). Paraku, sellisel viisil talitades alahindame varieeruvust, mistöttu meie statistiline test alahindab ebakindluse määra arvutatud statistiku ümber. Hierarhilised mudelid, mis kajastavad adekvatselt katse struktuuri, aitavad sellest murest üle saada. Üldine soovitus on, et kui teie katse struktuur seda võimaldab, siis peaksite alustama modelleerimist hierarhilistest mudelitest.

Mitmetasemelised mudelid on eriti kasulikud, kui teil on osades klastrites vähem andmepunkte kui teistes, sest nad vaatabad andmeid korraga nii klastrite vahel kui klastrite sees ning kannavad informatsiooni üle klastritest, kus on rohkem andmepunkte, nendesse klastritesse, kus on vähe andmeid. See töstab hinnangute täpsust.

Tavapärane regressioonimudel on sageli vaadeldav mitmetasandilise mudeli erijuuhuna. Näiteks kujutage ette mudelit, kus laste õppedukust on mõõdetud mitmes koolis. Kui gruppide (koolide) vaheline varieeruvus on väga madal, siis annab mitmetasemeline mudel sarnase tulemuse lihtsa mudeliga, kus kõik koolid on ühte patta kokku pandud. Ja vastupidi, kui koolid on üksteisest väga erinevad, siis võime sama hästi modelleerida iga kooli eraldi ja teistest sõltumatult. Samuti, kui meil on andmeid väga väheste koolide kohta, siis võib mitmetasandilsest mudelist saadav kasu olla tagasihoidlik, sest meil pole piisavalt andmeid, et modelleerida koolide vahelist varieeruvust. Samuti, kui meil on iga kooli kohta piisavalt palju andmeid, siis saame iga kooli eraldi modelleerides praktiliselt sama tulemuse kui mitmetasemelisest mudelist. Muudel juhtudel on töenäoliselt mõistlikum modelleerida õppedukust kahetasemelises mudelis, korraga õpilase tasemel ja kooli tasemel.

20.1 kahetasemeline mudel algebra keeles

1. tase on õpilse tase, 2. tase on klassi tase. Meil on j klassi, milles on erinev arv õpilasi. Iga õpilase kohta teame populaarsusindeksit (y - muutuja) ja sugu (x - muutuja). Klassi tasemel teame õpetaja staazi aastates (z - muutuja). Alustuseks on meil j regressioonivõrandit, eraldi võrrand igale klassile

$$y_{ij} = b_{0j} + b_{1j}X_{ij} + e_{ij}$$

(1. võrrand)

Kus subskript j tähistab klassi ja i tähistab õpilast. Näit b_{1j} tähendab, et me fitime igale klassile oma b_1 koefitsiendi (ja b_{0j} , et fitime igale klassile oma b_{0-i}). Me eeldame, et igal klassil on erinevad b_0 ja b_1 , mis tulevad vastavatest klassiülestest normaaljaotustest. Enamasti eeldame, et kõikide klasside varieeruvus on sama. Me modelleerime b_0 ja b_1 jaotusi, tuues sisse klassi tasemel muutuja z :

$$b_{0j} = \gamma_{00} + \gamma_{01}Z_j + u_{0j}$$

(2. võrrand)

$$b_{1j} = \gamma_{10} + \gamma_{11}Z_j + u_{1j}$$

(3. võrrand)

2. võrrand ennustab klassi keskmist populaarsusindeksit vastavalt õpetaja staazile.
3. võrrand ütleb, et populaarsuse ja soo seose tugevus sõltub õpetaja staazist – kui $\gamma_{11} > 0$, siis on seos seda tugevam, mida staazikam on õpetaja. u_{0j} ja u_{1j} on residuaalide vead klassi tasemel, mille kohta me eeldame, et

mean = 0 ja sõltumatust residuaalide vigadest õpilase tasemel (e_{ij}). u_{0j} residuaalide vigade dispersioon on σ_{u0}^2 jne. u_{0j} ja u_{1j} covariance on σ_{u01}^2 ja me ei eelda, et see = 0. Gamma koefitsiendid ei varieeru klasside vahel.

Asendades 1. võrrandis b_{0j} ja b_{1j} , saame oma võrrandisüsteemi muuta üheks pikaks võrrandiks.

$$Y_{ij} = \gamma_{00} + \gamma_{10}X_{ij} + \gamma_{01}Z_j + \gamma_{11}X_{ij}Z_j + u_{1j}X_{ij} + u_{0j} + e_{ij}$$

Siis näeme uues võrrandis liiget $\gamma_1 Z_j X_{ij}$, mis on interaktsioniliige. Ilma interaktsionita mudel näeb välja niimodi

$$Y_{ij} = \gamma_{00} + \gamma_{10}X_{ij} + \gamma_{01}Z_j + u_{1j}X_{ij} + u_{0j} + e_{ij}$$

juhusliku vealiige u_{ij} korrutub X_{ij} -ga, mistõttu sellest tulenev totaalne viga on erinev erinevatel X_{ij} väärustustel. Seega on meie mudel heteroskedastiline ja erineb selle pooltest tavalistest lineaarsetest mudelitest, mis eeldavad homoskedastilisust e residuaalvea sõltumatust X-i väärusest.

Teine oluline erinevus tavalisest lin mudelist on, et gupeeritud andmete puhul ei ole täidetud andmete iseseisvuse eeldus (gruppide sees modelleeritakse andmed korreleerituna - klassisisene korrelatsioon). Klassisisese korrelatsiooni saame intercept-only mudelist

$$Y_{ij} = \gamma_{00} + u_{0j} + e_{ij}$$

(selle mudeli saad, kui viskad pikast mudelist välja kõik X ja Z sisaldavad liikmed.) See mudel ajab varieeruvuse lahku kahe iseseisva komponendi vahel: σ_e^2 (1. taseme vigade dispersioon e_{ij}) ja σ_{u0}^2 (2. taseme vigade dispersioon u_{0j}).

Klassisisene korrelatsioon:

$$\rho = \frac{\sigma_{u0}^2}{\sigma_{u0}^2 + \sigma_e^2}$$

rho annab grupstruktuuri poolt seletatud variace proportsiooni, mida võib tõlgendada kui kahe sama gruvi juhusliku liikme vahelist oodatavat korrelatsiooni.

Üldiselt on kasulik töötada standardiseeritud regressioonikoefitsientidega, mida saab tõlgendada sd ühikutes. Erandiks on olukord, kus analüüs eesmärk on võrrelda erinevaid valimeid omavahel. Enamasti on kasulik standardiseerida andmed, mis mudelisse sisse lähevad, aga on võimalik ka standardiseerida koefitsiente kui selliseid.

$$\text{st_coef} = (\text{unstand_coef} \times \text{sd}(X)) / \text{sd}(Y)$$

Standardiseeritud andmete kasutamine muudab varieeruvuskomponentide fitte, aga jätab b koefitsientide fitid olemuselt samaks (see kehtib X-muutujate suvaliste lineaarsete transformatsioonide korral). Kui me jagame X-muutuja 2-ga, siis *uus* $b_1 = 2 \times \text{vana } b_1$. Mudeli poolt seletamata varieeruvuse proporsioon ei muutu. Eelnev kehtib senikaua, kuni mudeli tõusud (b_1) ei ole vabaks lastud, st need ei varieeru klassist klassi.

Tsentreerimine ($x_i - \text{mean}(x)$) mõjutab b_0 aga mitte b_1 koefitsiente, mis võib rääkida selle meetodi kasuks üle standardiseerimise ($\frac{x_i - \text{mean}(x)}{\text{sd}(x)}$), mis tekitab X muutujate jaotused, mille mean = 0 ja sd = 1.

NB! gruupi tasemel tsentreerimine, ehkki vahest kasulik, töötab hoopis teistmoodi kui üle kõikide gruppide tsentreerimine ja viib täiesti erineva mudelini - sellest tuleks hoiduda, senikaua kui te ei tea täpselt, mida te teete.

Mitmetasemelised mudelid on erilised, sest nad hõlmavad mitut varieeruvuse allikat, ei eelda konsantset vigade jaotust, ning modelleerivad seoseid erinevate mudeli tasemetega vahel.

20.1.1 Aegread

Aegridasid saab analüüsida mitmetasemilisena, kus korduvad mõõtmised (1. tase) on grupeeritud individide sisse (2. tase). Nii saab analüüsida ka ebaühtlase ajavahemiku järel tehtud mõõtmisi.

Aegridade analüüsiga lisaprobleem võrreldes tavalise mitmetasemelise mudeliga on, et me ei saa enam eeldada, et 1. taseme (individu) vead on üksteisest sõltumatud. Aegridade vaatluste vead on sageli ajas autokorreleeritud.

Y_{ti} - individu i õpitulemus ajapunktis t.

T_{ti} - ajapunkt

X_{ti} - ajas muutuv covariaat - kas õpilane töötab

Z_t - ajast mittesõltuv covariaat - sugu

1. tase (individu tase):

$$Y_{ti} = \pi_{0i} + \pi_{1i}xT_{ti} + \pi_{2i}X_{ti} + e_{ti}$$

2. tase (üle individide):

$$\pi_{0i} = \beta_{00} + \beta_{01}Z_t + u_{01}$$

$$\pi_{1i} = \beta_{10} + \beta_{11}Z_t + u_{11}$$

$$\pi_{2i} = \beta_{20} + \beta_{21} Z_t + u_{21}$$

Oluline punkt: aegridade modelleerimisel pakub meile sageli huvi ka ka korrelatsioon mudeli tõusu ja intercepti vahel. Kahjuks sõltub see näitaja sellest, millisest skaalast me ajamuutuja mudelisse sisse anname. Oluline on tagada, et ajaskaala nullpunkt on mõtekas.

To model growth - polynomial, logistic curve (first slow change, then quick, then slow again). Logistic parameters have meaning! Cubic polynomial approximates logistic and exponential curves - but here interpretation is on the level of some predicted growth curves.

<https://facebook.github.io/prophet> sessoonsete andmete fittimine ennustavasse mudelisse

<https://github.com/nwfsc-timeseries> kogu aegridade analüüsni pakette

```
library(coda)
#library(R2jags)
library(gridExtra)
library(broom)
```

20.1.2 Temporaalne autokorrelatsioon

eeldus: Mida lähemal on 2 ajapunkti üksteisele, seda suurem on nende vaheline korrelatsioon (ja residuaalide vaheline korrelatsioon). Tavaline lm eeldab, et see korrelatsioon = 0. Seda korrelatsiooni saab kas hinnata, või selle vastu võidelda. Meie teeme siin viimast.

Brms mudel näeb välja niimoodi

`cor_ar()` on brms funktsioon autokorrelatsiooni modelleerimiseks. selle formula on ühepoolne valem $\sim t$ või $\sim t | g$, mis annab ajakovariaadi t ja grupeeriva faktori g . Kui g on antud, siis modelleeritakse iga grupp iseseisvalt ja teistest gruppidest sõltumata (gruppide vahel on korrelatsioon 0).

Teine võimalus inkorporeerib mudelisse AR1 residuaalide autokorrelatsioonistruktuuri, kus korrelatsiooni eksponent väheneb ajas lineaarselt. Me eeldame, et see vähenemine toimub samamoodi sõltumata sellest, millises ajavahemikus me parasiagu oleme (statsionaarsuse eeldus).

Lihitsuse mõttes eeldame, et iga ajapunkti oodatud väärthus = tavaline lin prediktor + autokorrelatsiooni parameeter (ρ) korrutatuna eelmise vaatluse residuaaliga + tavapärase sõltumatu müra (σ^2).

20.2 mitmetasemeline mudel R-i mudelikeeles

Kui meie muutujad andmetabelis "data" on y = õpilase testiskoor, x = katsetingimus (binaarne faktor katse-kontroll, kalamaksaõli - platseebo), ja kool, siis "ühepajamudel" väljendub R-i mudelikeeles:

```
mudel <- lm(y ~ x, data=data)
```

ja mudel, kus iga kool on eraldi modelleeritud:

```
mudelid <- data %>% group_by(kool) %>% do(model = lm(y ~ x, data = .))
```

või purrr-i abil

```
data %>% split(.kool) %>% map(~ lm(y ~ x, data = .)) %>% map(summary)
%>% map_dfr(~ broom::glance(.), .id = "kool")
```

Seevastu hierarhiline mudel kirjutatakse kui

```
mudel <- lme4::lmer(y ~ x + (1 + x | kool), data=data)
```

või

```
mudel <- lme4::lmer(y ~ x + (1 | kool), data=data)
```

Esimesel juhul modelleeritakse igale koolile nii tõus kui intercept ja teisel juhul modelleeritakse igale koolile ainult intercept, seeläbi eeldades, et kõikidel koolidel on mudelis sama tõus, ehk kalamaksaõli efekt (ES = testitulemus kalamaksaõli grupis - testitulemus platseebogrupis). Intercept tähendab sellises mudelis enamasti baastaset (kontrolltingimus) ja tõus tähendab katseefekti (katsetingimus - kontrolltingimus). Seega eeldab teine mudel, et iga grupis võib küll olla oma baastase, aga katsefekt sellest ei muutu.

Lisaks, mudel

```
mudel <- lme4::lmer(y ~ x + (1 + x || kool), data=data)
```

modelleerib igale koolile tõusu ja intercepti lisaeeldusega, et tõusude ja interceptide vaheline korrelatsioon puudub. Ilma selle eelduseta piüüab mudel selle korrelatsiooni andmete põhjal leida. Kui andmeid on liiga vähe või mudel on liiga keeruline või korrelatsiooni võimalik esinemine tundub teadulsikult väga väheusutav, võib korrelatsiooni hindamisest loobuda, aga muidu tasub seda siiski hinnata.

```
mudel <- lme4::lmer(y ~ x + (1 + x | kool) + (1 + x | linn),
data=data)
```

Kui meil on mudelis rohkem kui 2 taset, kirjutame need sõltumata sellest, kas tasemed on hierarhiliselt üksteise sees (õpilane - kool - linn) või mitte (patsient - haigla - ravimi batch)

```
mudel <- lme4::lmer(y ~ x + (1 + x | grupp1) + (1 + x | grupp2),
data=data)
```

Kui esimesed 2 mudelit saab fittida lm() funktsiooniga, siis lihtne mitte-bayesiaanlik alternatiiv hierarhilise mudeli tarbeks on lme4 pakett (<https://cran.r-project.org/web/packages/lme4/vignettes/lmer.pdf>), mis on lihtsam, kiirem ja ebatäpsem *ad hoc* viis arvutada mitmetasemeliseid mudeliteid, kui Stan. Selle eelis Stani ees on eelkõige kiirus ja puuduseks on väiksem paindlikus mudelite formulierimisel ja see, et väikeste valimite ja vähestega gruppide puhul töötab lme4 algoritm palju halvemini, kui bayesi lahendused. Seega kasutame me pigem Stani, kui lme4. Samas, suurepärane pakett nimega brms (https://cran.r-project.org/web/packages/brms/vignettes/brms_overview.pdf) suudab tõlkida lme4 mudeli kirjelduse otse Stani keelde ja seda mudelit seal jooksutada. Brms teeb elu magusaks (vt lisa 2).

Suurem sõltuvus valimi suurusest ja erinevatest lisaeeldustest võrreldes Bayesi mudelitega on see hind, mida ad hoc lahendused maksavad oma lihtsuse eest. Enamust klassikalisi teste (t test, chi ruut test, jms) võib vaadelda selliste ad hoc lahendustena, mis sageli lagunevad laialt väikesetel valimitel, samas kui bayes töötab väikeste valimitega hästi – tõsi küll, sõltudes väikeste valimite korral rohkem priorist ja andes seal realistikult laiad usaldusintervallid.

20.3 Mitmetasemeliste mudelite lisaeeldused

Mitmetasandilised mudelid toovad sisse lisaeelduse, et lineaarsuse/normaalsuse jm eeldused kehtivad igal mudeli tasemel. Samuti, et kõik grupid tulevad samast statistilisest populatsionist, ja vastavalt sellele on nad mudelis koondatud ühise priori alla. Mitmetasemelises mudelis töötab gruvi tasemel mudel priorina indiviidi tasemel mudelile.

20.4 Mitmetasemeline mudel töötab korraga mitmel tasmel

Mudeli muudab mitmetasemeliseks see, et me määrame veamudelit kasutades mitte ainult indiviidi tasmel koefitsiente (1. tase), vaid anname neile koefitsientidele omakorda veamudeli (2. tase), mis modelleerib koefitsientide varieeruvust gruppide vahel. Selliseid tasemeid võib lisada põhimõtteliselt ükskõik kui palju.

Kõrgema taseme mudel, lisaks sellele, et modelleerida gruppide vahelist varieeruvust, töötab ka priorina madalama taseme suhtes. Seega saab mudeli fittimisel 2. tase informatsiooni 1. tasemelt (andmete näol) ja samal ajal annab informatsiooni esimesele tasemele (priori kujul).

1. Mitmetasemelised mudelid modelleerivad eksplitsiitselt varieeruvust klasriste sees ja klastrite vahel.
2. Nad modelleerivad indiviidi-tasemel regressioonikoeffitsientide varieeruvust.
3. Nad võimaldavad paremini määrata indiviidi tasemel regressioonikoeffsiente endid, eriti kui erinevates gruppides on erinev arv indiviide.

Shrinkage

Oletame, et te plaanite reisi Kopenhaagenisse ja soovite sellega seoses teada, kui kallis on keskelt läbi ölu selle linna kõrtsides. Teile on teada ölle hind kolmes Kopenhaageni kõrtsis, mida ei ole just palju. Aga sellele lisaks on teile teada ka ölle hind 6-s Viini, 4-s Praha ja 5-s Pariisi kõrtsis. Nüüd on teil põhimõtteliselt kolm võimalust, kuidas sellele probleemile läheneda.

1. Te arvestate ainult Kopenhaageni andmeid ja ignoreerite teisi, kui ebarelevantseid. See meetod töötab hästi siis, kui teil on Kopenhaageni kohta palju andmeid (aga teil ei ole).
2. Te arvestate võrdsest kõiki andmeid, mis teil on — ehk te võtate keskmise kõikidest õllehindadest, hoolimata riigist. See töötab parimini siis, kui pärисelt pole vahet, millisest riigist te oma ölle ostate, ehk kui ölu maksab igal pool sama palju. Antud juhul pole see ilmselt parim eeldus.
3. Te eeldate, et ölle hinna kujunemisel erinevates riikides on midagi ühist, aga et seal on ka erinevusi. Sellisel juhul tahate te fittida hierarhilise mudeli, kus teie hinnang ölle hinnale Kopenhaagenis sõltuks mingil määral (aga mitte nii suurel määral, kui eelmises punktis) ka teie kogemustest teistes linnades. Sama moodi, teie hinnang ölle hinnale Pariisis, Prahas jne hakkab mingil määral sõltuma kõikide linnade andmetest.

Kui teil on olukord, kus te mõõdate erinevaid gruppe, mis küll omavahel erinevad, aga on ka teatud määral sarnased (näiteks testitulemused grupeerituna kooli kaupa), siis on mõistlik kasutada kõikide gruppide andmeid, et adjussteerida iga gruubi spetsiifilisi parameetreid. Seda adjussteerimise määra kutsutakse “shrinkage”.

Shrinkage toimub parameetri keskväärtuse suunas ja mingi gruubi shrinkage on seda suurem, mida vähem on selles gruubis liikmeid ja mida kaugemal asub see gruub kõikide gruppide keskväärtusest. See viib shrinkage koefitsientide kallutatusele (bias), aga samas ka suuremale täpsusele (precision). See tähendab, et shrinkage koefitsiendi hinnang on keskelt lähemal töelisele koefitsiendi värtusele kui hinnang tavaliise ühetasandilise mudeli koefitsiendile.

20.4. MITMETASEMELINE MUDEL TÖÖTAB KORRAGA MITMEL TASMEL247

Shrinkage on põhimõtteliselt sama nähtus, mis juba Francis Galtoni poolt avastatud regressioon keskmisele. Regressioon keskmisele on stohhastiline protsess kus, olles sooritanud n mõõtmist ja arvutanud nende tulemuste põhjal efekti suuruse, see valimi ES peegeldab nii tegelikku ES-i kui juhuslikku valimiviga. Kui valimivea osakaal ES-s on suur, siis lisamõõtmised vähendavad keskeltläbi efekti suurust. Shrinkage erineb sellest ainult selle poolest, et lisamõõtmised meenutavad ainult **osaliselt** algseid mõõtmisi.

Kasutades hierarhilisi mudeliteid saab võidelda ka valehäirete ehk mitmese testimise probleemiga. See probleem on lihtsalt sõnastatav: kui te sooritate palju võrdluskatseid ja statistilisi teste olukorras, kus tegelik katseefekt on tühine, siis tänu valimiveale annavad osad teie paljudest testidest ülehinnatud efekti. Seega, kui meil on kahtlus, et enamus võrdlusi on "mõttetud" ja me ei oska ette ennustada, millised võrdlused neist (kui üldse mõni) võiks anda töelise teaduslikult mõttuka efekti, siis on lahendus kõiki saadud efekte kunstlikult pisendada kõikide efektide keskmise suunas. Mudeli kontekstis kutsutakse sellist lähenemist *shrinkage*-ks. Aga kui suurel määral seda teha? See sõltub nii sellest, kui palju teste me teeme, valimi suurusest, kui ka sellest, kuidas jaotuvad mõõdetud efektisuurused (milline on efektisuuruste varieeruvus testide vahel).

Bayesi lahendus on, et me lisame mudelisse veel ühe hierarhilise priori, mis kõrgub üle gruppide-spetsiifilise priori. Seega anname me olemasolevale priorile uue kõrgema taseme meta-priori, mis tagab, et informatsiooni jagatakse gruppide vahel ja samal ajal ka gruppide sees. Sellise lahenduse õigustus on, et me usume, et erinevad alam-grupid pärinevad samast üli-jaotusest ja neil on omavahel midagi ühist (ehkki alam-gruppide vahel võib olla ka reaalseid erinevusi). Näiteks, et kõik klassid saavad oma lapsed samast lastepopulatsionist, aga siiski, et leidub ka eriklassse eriti andekatele.

Selline mudel tagab, et samamoodi nagu mudeli ennustused individuaalsete andmepunktide kohta iga alam-gruppi sees "liiguvald lähemale" oma alam-gruppi keskmisele, samamoodi liiguvald ka alam-gruppide keskmised lähemale üldisele gruppi keskmisele. Selle positiivne mõju on valealarmide vähendamine ja oht on, et me kaotame ka töelisi efekte. Bayesi eelis on, et see oht realiseerub ainult niipalju, kuipalju meie mudel ei kajasta reaalset katse struktuuri. Klassikalises statistikas rakendataavad multiple testingu korrektsioonid (Bonferroni, ANOVA jt) on kõik teoreetiliselt kehvemad.

Lihtsaim shrinkage mudeli tüüp on mudel, kus me laseme vabaks interceptid, aga mitte tõusunurgad. Igale klastrile vastab mudelis oma intercepti parameeter ja oma intercepti prior. Lisaks annab mudel meile fittimise käigus valimi andmete põhjal ise parameetrid kõrgema taseme priorisse, mis on ühine kõikidele interceptidele. Seega me määrame korraga interceptide parameetrid ja kõrgema taseme priori parameetrid, mis tähendab, et informatsioon liigub mudelit fittides mõlemat pidi — mõöda hierarhiat alt ülesse ja ülevalt alla. Selline mudel usub, et erinevate koolide keskmise tase erineb (seda näitab iga kooli intercept), aga juhul kui me mõõdame näiteks kalamaksaõli mõju õppedukusele, siis selle mõju suurus ei erine koolide vahel (kõikide koolide tõusuparameetrid on identsed).

Shrinkage kui nähtuse avastas Francis Galton 1870-ndatel aastatel. Galton ja tema sõbrad veetsid nimelt kümme aastat üle Inglismaa taimi kasvatades ja mõõtes erinevate põlvkondade seemnete suurusi. Eesmärk oli luua tühjale kohale uus teadus, pidevate tunnuste geneetika, ja katsete tulemus oli rabav. Nimelt leiti tugev seaduspära, mille kohaselt suurte seemnetega emataimedest türed andsid keskeltläbi väiksemaid seemneid kui nende vanemad ja vastupidi, väikeste seemnetega emade türed andsid keskeltläbi suuremaid seemneid. Galtoni usk, et ta on avastanud tähtsa bioloogiaseaduse, purunes ca 1885, kui ta pääsес analüüsima tuhatkonna inimese pikkusi andmestikus, mis sisaldas vanemate ja täiskasvanud laste pikkusi, ning leidis seal sama nähtuse. Sertifitseeritud geeniusena mõistis Galton, et ta ei olnud avastanud mitte niivõrd geneetikaseaduse, vaid peaaegu, et loogikaseaduse. Tema enda sõnadega: The average regression of the offspring to a constant fraction of their mid-parental deviations, is now shown to be a perfectly reasonable law which might have been deductively foreseen. It is of so simple a character that I have made an arrangement with pulleys and weights by which the probable average height of the children of known parents can be mechanically reckoned. (vt joonis). Sellega avastas Galton regressiooni keskmisele, mis on sisuliselt sama asi, mis shrinkage. Galton nägi, et shrinkagel on järgmised omadused: 1. “The mean filial regression towards mediocrity was directly proportional to the parental deviation from it.” Ehk, mida kaugemal on vanemad keskmisest, seda suurema amplituudiga on nende laste shrinkage 2. “The child inherits partly from his parents, partly from his ancestry. . . . the further his genealogy goes back, the more numerous and varied will his ancestry become . . . Their mean stature will then be the same as that of the race; in other words, it will be mediocre.” Ehk, shrinkage toimub alati, kui tunnuse väärthus ei ole deterministlikult määratud (shrinkage taandub korrelatsioonile vanemate ja laste varieeruvuse vahel) 3. “This law tells heavily against the full hereditary transmission of any gift. The more exceptional the amount of the gift, the more exceptional will be the good fortune of a parent who has a son who equals him in that respect.” Ehk, pidevate tunnuste korral on korrelatsioon alati $<1 & >1$ 4. “The law is even-handed; it levies the same heavy succession-tax on the transmission of badness as well as of goodness. If it discourages the extravagant expectations of gifted parents that their children will inherit all their powers, it no less discountenances extravagant fears that they will inherit all their weaknesses and diseases.” Ehk shrinkage töötab võrdselt mõlemas suunas (ülevalt alla ja alt üles, aga ka vanematele lastele ja lastelt vanematele). Seega ei ole shrinkage ajas toimuv, põhjuslik, ega isegi mitte füüsikaline protsess, vaid tõenäosusteooriast tulenev loogiline paratamatus. Samamoodi nagu shrinkage esineb vanemate-laste vahel esineb see ka valimi-kordusvalimi vahel kõigi valimite

keskmise suunas (valimiefektid taanduvad välja sedamõõda, kuidas valimeid juurde tuleb). Ja samamoodi, kui me võtame valimi testitulemusi mitmest koolist, siis eeldusel, et õpilased on kõikides koolides sarnased (aga mitte identsed), toimub shrinkage kõikide koolide keskmise suunas. Seega, nihutades mingi kooli keskmist testitulemust koolide keskmise suunas, saame parema hinnangu selle kooli õpilaste teadmisetele kui pelgalt selles koolis õpilaste teadmisi mõõtes!

20.5 ANOVA-laadne mudel

Lihtne ANOVA on sageduslik test, mis võrdleb gruppide keskmisi mitmese testimise kontekstis. Siin ehitame selle Bayesi analoogi, mis samuti hindab gruppide keskmisi mitmese testimise kontekstis. Põhiline erinevus seisneb selles, et kui ANOVA punktennustus iga gruvi keskväärtusele võrdub valimi keskväärtusega ja ANOVA pelgalt kohandab usaldusintervalle selle keskväärtuse ümber, siis bayesianlik mudel püüab ennustada igale grupile selle tegelikku kõige tõenäolisemat keskväärtust arvestades kõigi gruppide andmeid. Shrinkage-i roll on ekstreemseid gruppe "tagasi tömmates" vähendada ebakindlust iga gruvi keskmise ennustuse ümber. Shrinkage käigus tömmatakse grupee kõikide gruppide keskmise poole seda tugevamalt, mida kaugemal nad sellest keskmisest on. Sellega kaasneb paratamatult mõningane süstemaatiline viga, kus tõelised efektid tulevad välja väiksemata, kui nad tegelikult on. Kui ilma tegelike efektidega gruppide arv on väga suur võrreldes päris efektidega gruppidega, siis võib shrinkage meie pärisefektid sootuks ära kaotada. Kahjuks on see loogiline paratamatus; alternatiiviks on olukord, kus meie üksikud pärisefektid upuvad sama suurte pseudoefektide merre.

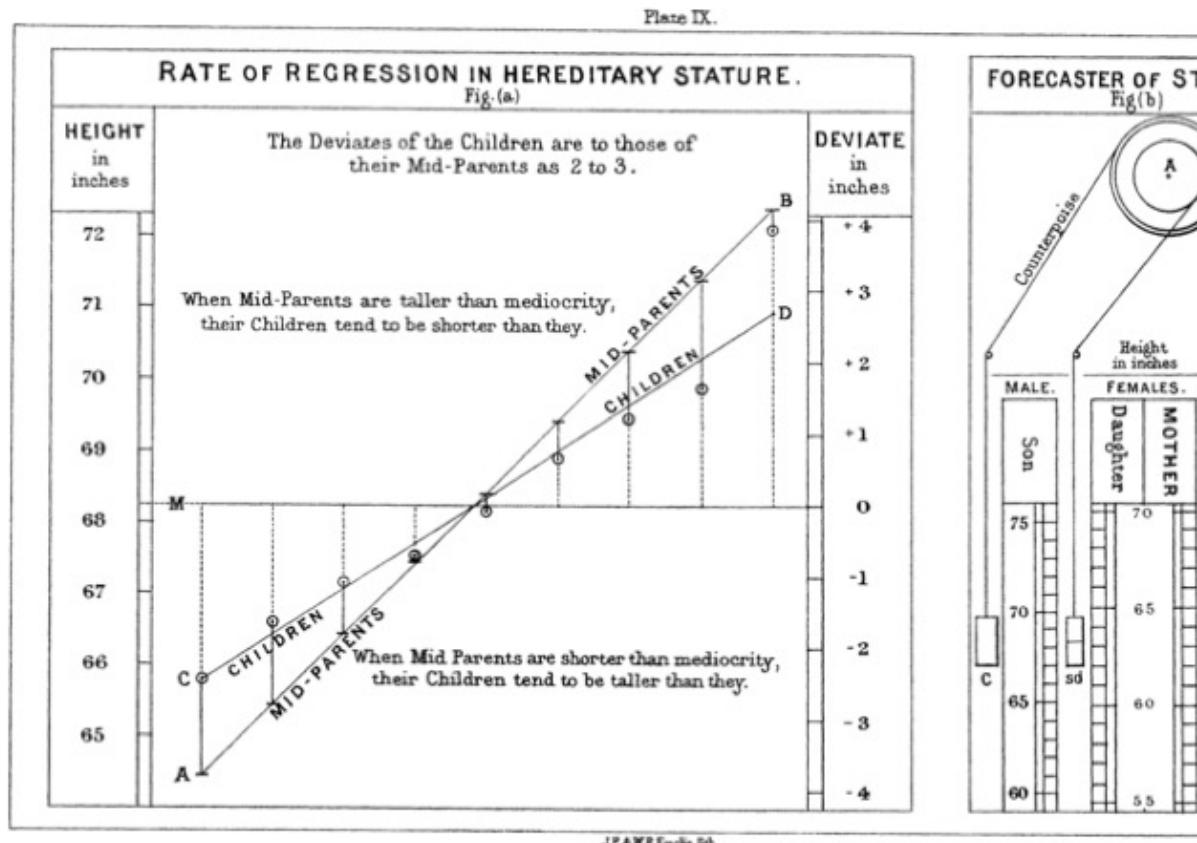
Ok, aitab mulast, laadime vajalikud raamatukogud ja andmed ning vaatame mis saab.

```
library(tidyverse)
library(stringr)
library(rethinking)
library(psych)
```

Andmed: *The data contain GCSE exam scores on a science subject. Two components of the exam were chosen as outcome variables: written paper and course work. There are 1,905 students from 73 schools in England. Five fields are as follows.*

1. School ID
2. Student ID
3. Gender of student

0 = boy



1 = girl

4. Total score of written paper
5. Total score of coursework paper

Missing values are coded as -1.

```
schools <- read_csv( "data/schools.csv")
schools <- schools %>%
  filter(complete.cases(.)) %>%
  mutate_at(vars(sex, school), as.factor)
## map2stan requires data.frame
class(schools) <- "data.frame"
```

Alustuseks mitte-hierarhilise mudel, mis arvutab keskmise score1 igale koolile eraldi. See on intercept-only mudel, mis tähendab, et me hindame testitulemuse keskväärtust kooli kaupa ja igale koolile sõltumatult kõigist teistest koolidest. Me ei püüa siin ennustada testitulemuste väärtsusi x-i väärtsuste põhjal. Selles mudelis on tavapärased ühetasemelised priorid, ainult mu on ümber nimetatud a_schooliks ja sellele on antud indeks [school], mis tähendab, et mudel arvutab a_school-i, ehk keskmise testitulemuse, igale koolile. Kuna siin puuduvad kõrgema taseme priorid, siis vaatab mudel igat kooli eraldi ja ühegi kooli hinnang ei arvesta ühegi teise kooli andmetega.

```
schoolm2 <- map2stan(
  alist(
    score1 ~ dnorm(mu, sigma),
    mu <- Intercept + v_Intercept[school],
    Intercept ~ dnorm(0, 50),
    v_Intercept[school] ~ dnorm(0, 50),
    sigma ~ dcauchy(0, 2)
  ), data = schools)
```

Vaata koefitsente.

```
precis(schoolm2, depth = 2)
```

Igale koolile antud hinnang on sõltumatu kõigist teistest koolidest.

Ja nüüd hierarhilise mudel, mis teab koolide vahelisest varieeruvusest. Siin leibab a_school-i priorist teise taseme meta-parameetri nimega sigma_school, millele on defineeritud oma meta-prior.

```
schoolm3 <- map2stan(alist(
  score1 ~ dnorm(mu, sigma),
  mu <- Intercept + v_Intercept[school],
  Intercept ~ dnorm(0, 50),
  v_Intercept[school] ~ dnorm(0, sigma_school),
  sigma_school ~ dcauchy(0, 2),
```

```
sigma ~ dcauchy(0, 2)
), data = schools)

precis(schoolm3, depth = 2)
```

Nagu näha on `sigma_school < sigma`, mis tähendab, et koolide vaheline varieeruvus on väiksem kui õpilaste vaheline varieeruvus neis koolides. Seega sõltub testi tulemus rohkem sellest, kes testi teeb kui sellest, mis koolis ta käib. Loogika on siin järgmine: samamoodi nagu testitulemustel on jaotus õpilasekaupa, on neil ka jaotus koolikaupa. Koolikaupa jaotus töötab priorina õpilasekaupa jaotusele. Aga samas vajab kooli kaupa jaotus oma priorit — ehk meta-priorit. Seega saame me samast mudelist hinnangu nii testitulemustele kõikvõimalike õpilaste lõikes, kui ka kõikvõimalike koolide lõikes. Mudel ennustab ka nende koolide ja õpilaste tulemusi, keda tegelikult olemas ei ole, aga kes võksid kunagi sündida.

Ning veel üks hierarhiline mudel, mis teab nii koolide skooride keskmiste varieeruvust kui koolide vahelist varieeruvust.

Võrdleme mudeleid.

```
compare(schoolm2, schoolm3)
```

Sit nähtub, et m3 on parim mudel, aga ka m2 omab mingit kaalu.

```
coeftab_plot(coeftab(schoolm2, schoolm3), cex = 0.5)
```

Siin on hästi näha shrinkage m3 puhul võrreldes m2-ga, mis ei tee multiple testingu korrektsiooni. Nende koolide puhul, kus usaldusintervall on laiem, on ka suurem shrinkage (mudel võtab nende kohta suhteliselt rohkem infot teistest koolidest sest need koolid ise on mingil põhjusel suhteliselt infovaesed).

20.6 Vabad interceptid klassikalises regres-sioonimudelis

Ennustame score1 sõltuvust sex-ist. Küsimus: kui palju poiste ja tüdrukute matemaatikaoskused erinevad? Fitime mudeli, mis laseb vabaks intercepti. **Selle mudeli eeldus on, et igal koolil on oma baastase (oma intercept), aga kõikide koolide efektid (mudeli tõusu-koeefitsient) on identsed.**

```
describe(schools)
#>      vars     n    mean      sd median trimmed   mad   min   max range
#> school*     1 1523  38.36  19.98   40.0   38.84  23.7 1.00    73  72.0
#> student     2 1523 1016.45 1836.14  129.0  628.65 124.5 1.00 5516 5515.0
#> sex*        3 1523    1.59    0.49    2.0     1.61    0.0 1.00     2    1.0
#> score1      4 1523   46.50   13.48   46.0   46.68  13.3 0.60    90  89.4
#> score2      5 1523   73.38   16.44   75.9   74.65  16.5 9.25   100  90.8
```

20.6. VABAD INTERCEPTID KLAASSIKALISES REGRESSIOONIMUDELIS 253

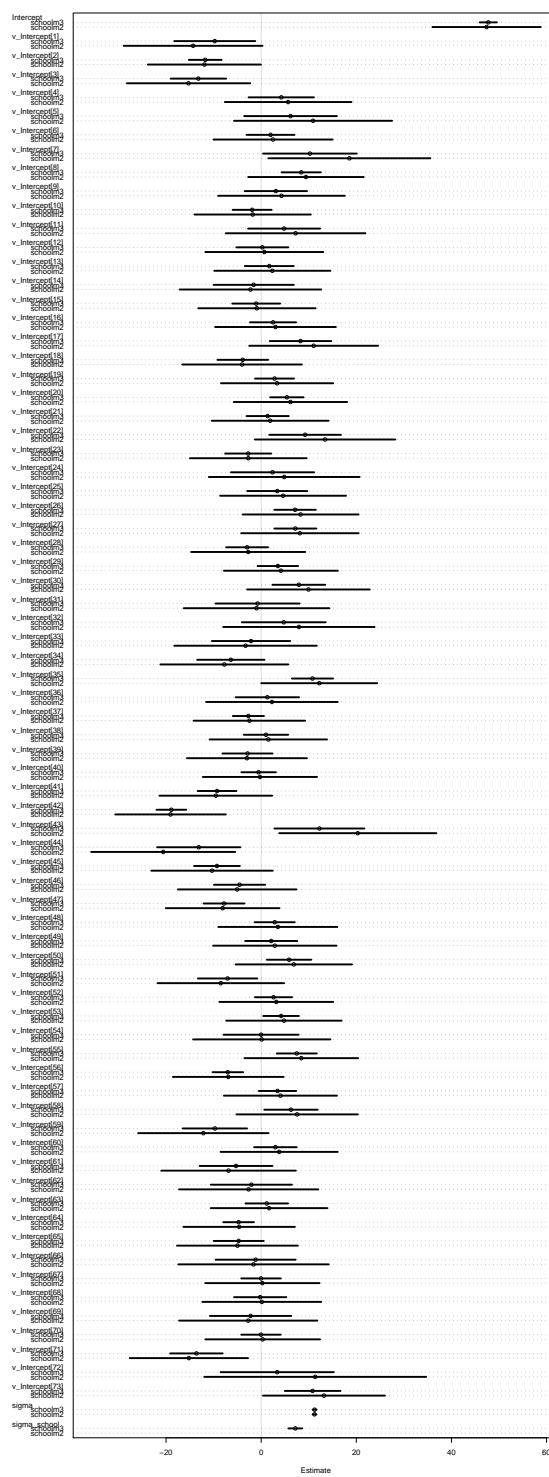


Figure 20.2: Mudelite koefitsiendid.

```
#>      skew kurtosis    se
#> school* -0.18    -1.08  0.51
#> student   1.69     0.98 47.05
#> sex*      -0.37    -1.87  0.01
#> score1   -0.12    -0.05  0.35
#> score2   -0.75     0.51  0.42
```

Me kasutame prediktorina binaarset kategoorilist muutujat. See on analoogiline olukord ANOVA mudelile, mis võtab arvesse multiple testingu olukorra, mis meil siin on.

```
schools_f1 <- glimmer(score1 ~ sex + (1 | school), data = schools)
#> alist(
#>   score1 ~ dnorm(mu , sigma ),
#>   mu <- Intercept +
#>     b_sex1*sex1 +
#>     v_Intercept[school],
#>   Intercept ~ dnorm(0,10),
#>   b_sex1 ~ dnorm(0,10),
#>   v_Intercept[school] ~ dnorm(0,sigma_school),
#>   sigma_school ~ dcauchy(0,2),
#>   sigma ~ dcauchy(0,2)
#> )
```

Kuna glimmeri priorite parametriseringud on vales skaalas (liiga väikesed), muudame neid nii, et intercept (keskmine testitulemus üle koolide) oleks tsentreeritud 50-le (max testi tulemus on 100) ja standardhälve on 20. Igaks juhusks tõstame veidi ka beta koefitsiendi priori sigmat. v_intercept peaks olema alati nullile tsentreeritud.

Glimmeri väljundis on sama palju koolide veerge, kui palju on erinevaid koole, miinus üks. Selline binaarne numbriline väljund on Stani-le vajalik. Seega ei saa me faktortunnuste korral kasutada algset andmetabelit.

```
head(schools_f1$d)

schools_m1 <- map2stan(alist(
  score1 ~ dnorm(mu , sigma ),
  mu <- Intercept + b_sex1*sex1 + v_Intercept[school],
  Intercept ~ dnorm(50, 20),
  b_sex1 ~ dnorm(0, 15),
  v_Intercept[school] ~ dnorm(0, sigma_school),
  sigma_school ~ dcauchy(0,2),
  sigma ~ dcauchy(0,2)
), data = schools_f1$d) # use the data table generated by glimmer()
#glimmer converts factors to Stan-eatable form.
```

Siin on v_Intercept kooli-spetsiifiline korrektsioonifaktor, mis tuleb liita üld-

20.6. VABAD INTERCEPTID KЛАSSIKALISES REGRESSIOONIMUDELIS 255

isele Interceptile. `mean(v_Intercept) == 0`. Me eeldame, et korrektsioonid on normaaljaotusega. Alternatiivne viis seda mudelit kirjutada oleks `mu <- Intercept[school] + b_sex1*sex1` ja see töötab smamoodi (nүüd on iga kooli intercept kohe eraldi).

```
plot(precis(schools_m1, depth = 2), cex = 0.5)

precis(schools_m1)
#>           Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
#> Intercept 49.21  0.98   47.89    51.01   227    1
#> b_sex1    -2.44  0.60   -3.37   -1.52   1000    1
#> sigma_school 7.06  0.71   5.87    8.09   1000    1
#> sigma     11.18  0.20  10.85   11.50   1000    1
```

`sex = 1` ehk `sex1` on tüdruk.

Intercept annab siin `sex = 0` (poisid) keskmise skoori kooli kaupa (kui liita üldisele interceptile kooli-spetsiifiline intercept). Kui tahame näiteks hinnangut 2. kooli tüdrukute skoorile (ehk töelisele matemaatikavõimeküsele) siis:

```
Intercept + b_sex1 + intercept[2]
```

annab meile selle posteeriorigi. Poistele sama 2. kooli kohta:

```
Intercept + intercept[2]
```

Ja poiste-tüdrukute erinevus skooripunktides võrdub

```
b_sex1
```

Arvutame siis kooli nr 2 tüdrukute keskmise skoori posteeriorigi.

```
schools_m1_samples <- as.data.frame(schools_m1@stanfit)
school_2_girls <- schools_m1_samples$Intercept +
  schools_m1_samples$b_sex1 +
  schools_m1_samples$v_Intercept[2]
## Plot density histogram of intercepts
dens(school_2_girls)
```

Ja Poiste oma

```
school_2_boys <- schools_m1_samples$Intercept +
  schools_m1_samples$v_Intercept[2]
## Plot density histogram of intercepts
dens(school_2_boys)
```

Siin on eeldus, et kõikides koolides on sama poiste ja tüdrukute vaheline erinevus (`b_sex1`), kuid erinevad matemaatikateadmiste baastasemed (mudeli intercept on koolide vahel vabaks lastud, kuid tõus mitte).

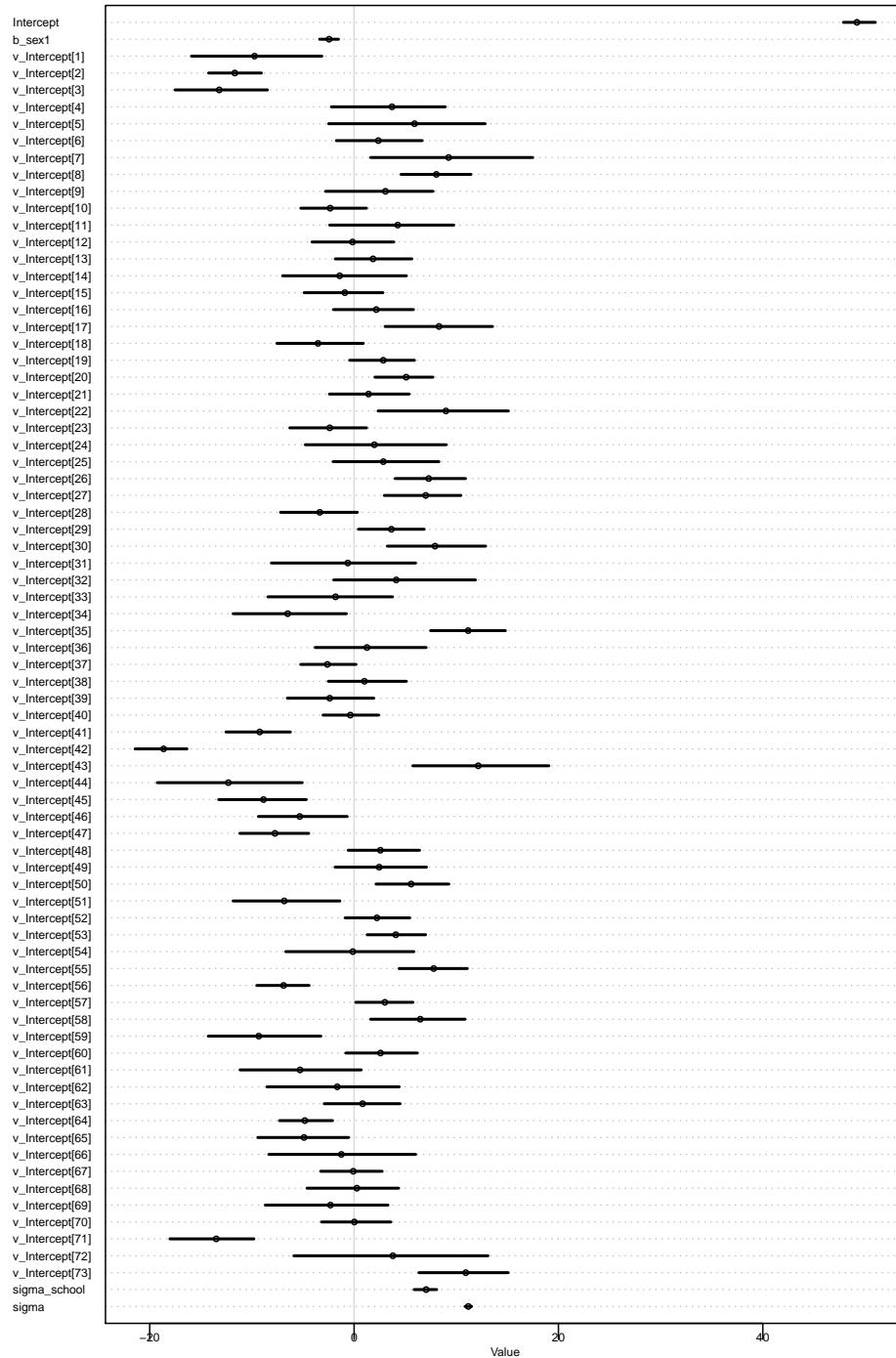


Figure 20.3: Mudeli koefitsiendid

20.6. VABAD INTERCEPTID KЛАSSIKALISES REGRESSIOONIMUDELIS 257

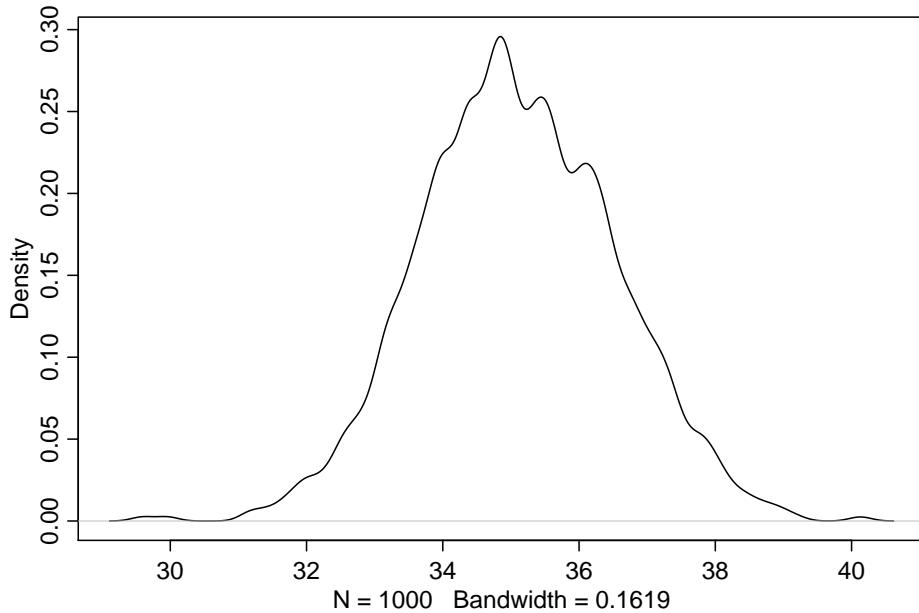


Figure 20.4: Tüdrukute skoori posteerior

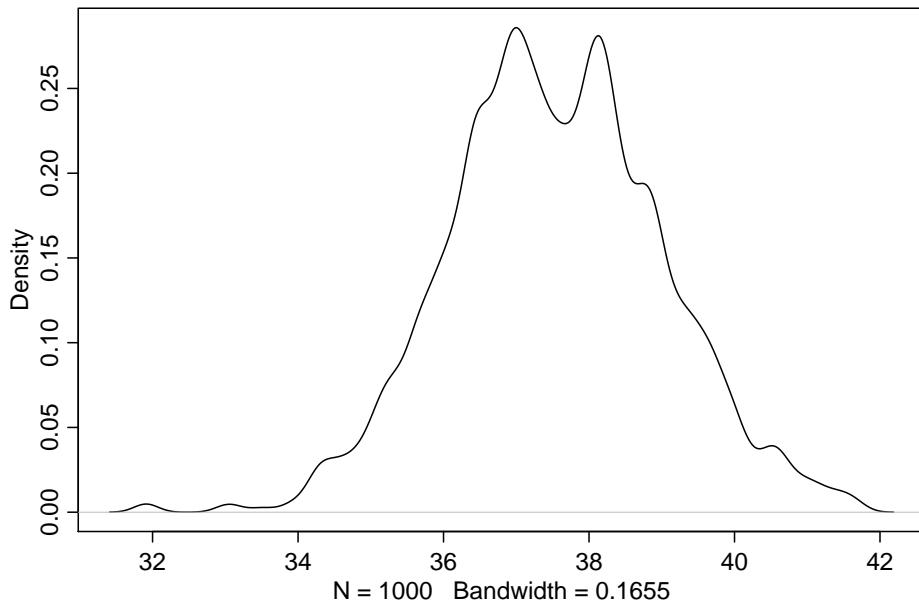


Figure 20.5: Poiste skoori posteerior.

20.7 Vabad tõusud ja interceptid

Milline näeb välja mudel, kus me laseme vabaks nii intercepti kui tõusu?

```
schools_f2 <- glimmer(score1 ~ sex + (1 + sex | school), data = schools)
#> alist(
#>   score1 ~ dnorm(mu, sigma),
#>   mu <- Intercept +
#>     b_sex1*sex1 +
#>     v_Intercept[school] +
#>     v_sex1[school]*sex1,
#>   Intercept ~ dnorm(0, 10),
#>   b_sex1 ~ dnorm(0, 10),
#>   c(v_Intercept, v_sex1)[school] ~ dmvnorm2(0, sigma_school, Rho_school),
#>   sigma_school ~ dcauchy(0, 2),
#>   Rho_school ~ dlkjcorr(2),
#>   sigma ~ dcauchy(0, 2)
#> )
```

Nüüd on meil lisaparametrid v_sex1, mis annab tõusu igale koolile eraldi ning Rho_school, mis annab korrelatsiooni intercepti ja tõusu vahel. Nüüd me jagame informatsiooni erinevat tüüpi parametreite, nimelt interceptide ja tõusude, vahel. Selleks ongi vaja Rho lisa-parametrit. Nüüd ei modelleeri me intercepti ja tõusu enam 2 eraldi normaaljaotuste abil vaid ühe 2-dimensionaalse normaaljaotusega (mvnorm2).

Prior korrelatsioonile Interceptide ja tõusude vahel on `rthinking::lkjcorr()`. Selle ainus parameeter on K. Mida suurem K, seda rohkem on prior konsentreeritud 0 korrelatsiooni ümber. K = 1 annab tasase priori. Meie kasutame K = 2, mis töötab laia vahemiku mudelitega.

```
R <- rlkjcorr(1e4, K = 2, eta = 2)
dens(R[, 1, 2], xlab = "correlation")

schools_m2 <- map2stan(alist(
  score1 ~ dnorm(mu, sigma),
  mu <- Intercept +
    b_sex1*sex1 +
    v_Intercept[school] +
    v_sex1[school]*sex1,
  Intercept ~ dnorm(50, 20),
  b_sex1 ~ dnorm(0, 20),
  c(v_Intercept, v_sex1)[school] ~ dmvnorm2(0, sigma_school, Rho_school),
  sigma_school ~ dcauchy(0, 2),
  Rho_school ~ dlkjcorr(2),
  sigma ~ dcauchy(0, 2)
), schools_f2$d)
```

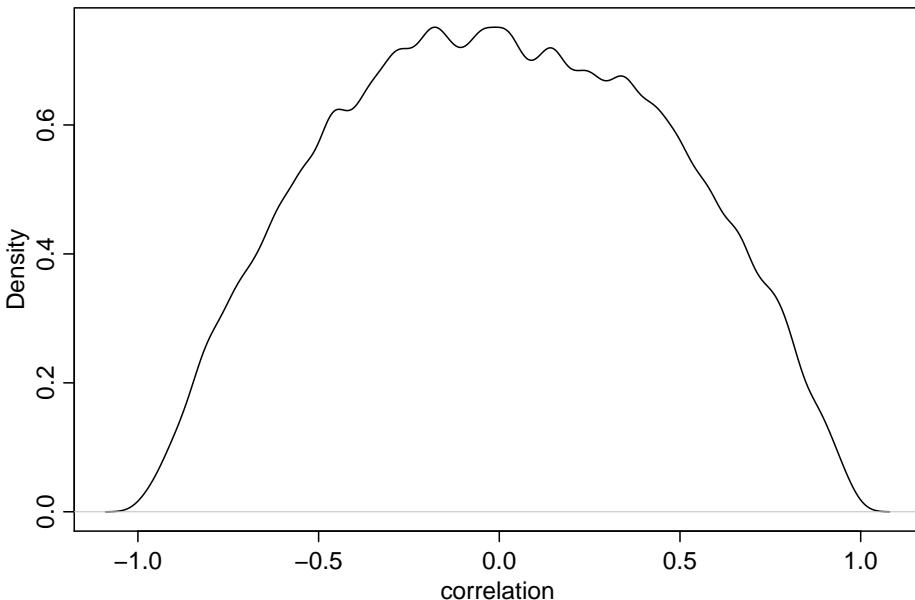


Figure 20.6: Korrelatsiooni prior on nõrgalt informatiivne – suunab posteeriori eemale ekstreemsetest korrelatsioonidest.

```
plot(precis(schools_m2, depth = 2), cex = 0.5)
```

Posteerior korrelatsioonile intercepti ja tõusu vahel:

```
schools_m2_samples <- extract.samples(schools_m2)
df1 <- schools_m2_samples$Rho_school %>% as.data.frame()
#df1 #corr matrix- we need only the V2 col
dens(df1$V2)
```

Meil on negatiivne korrelatsioon intercepti ja tõusu vahel. Seega, mida väiksem on poiste keskmise skoor koolis (=intercept), seda suurem om erinevus poiste ja tüdrukute skooride vahel (=tõus).

Nüüd saab 2. kooli skoori tüdrukutele valemiga:

$$\text{Intercept} + b_{\text{sex1}} + v_{\text{intercept}[2]} + v_{\text{sex1}[2]}$$

Sama skoor poistele:

$$\text{Intercept} + v_{\text{intercept}[2]}$$

ja tüdrukute ja poiste erinevus 2. koolile:

$$b_{\text{sex1}} + v_{\text{sex1}[2]}$$

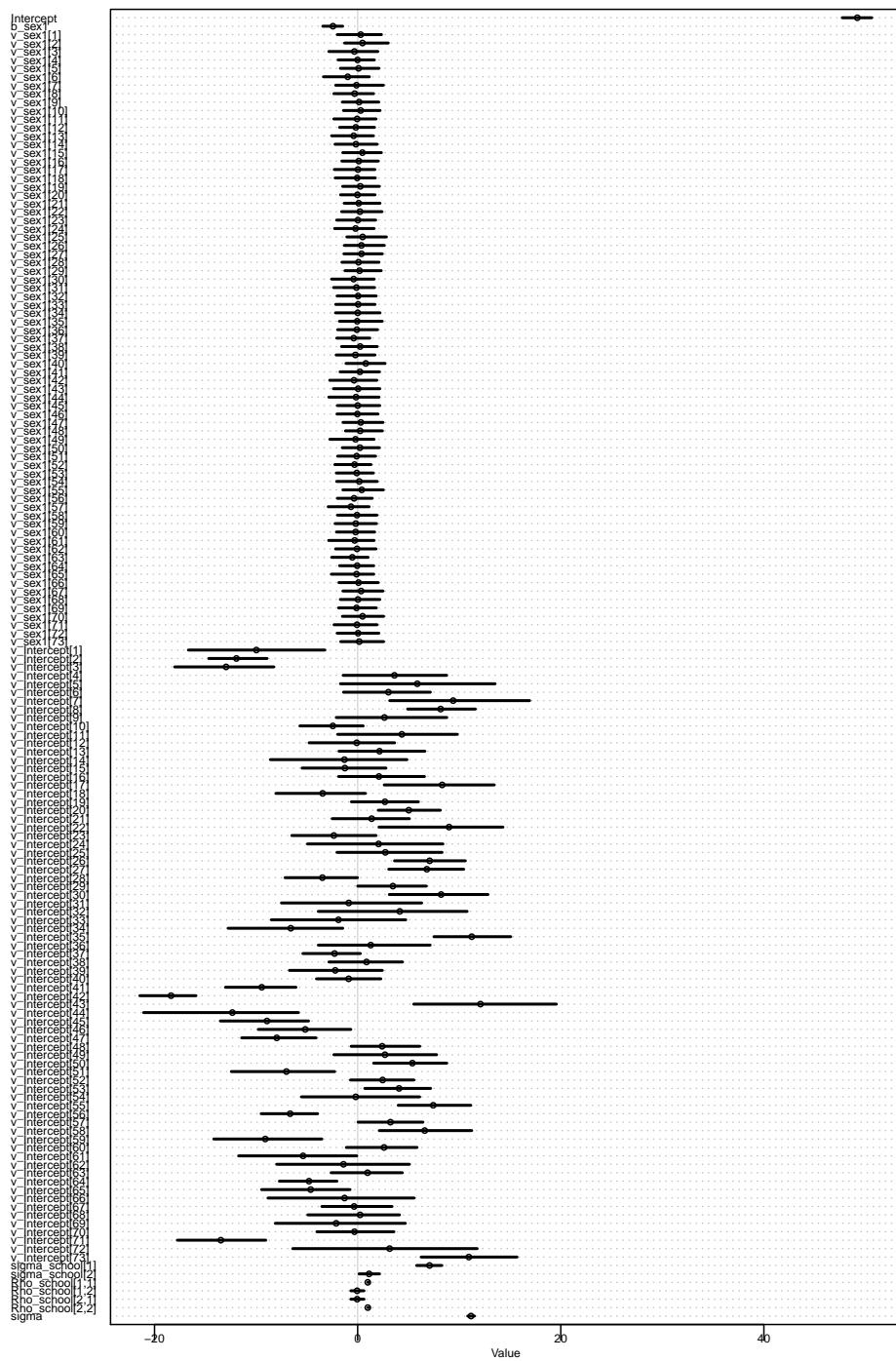


Figure 20.7: Mudeli m2 koefitsiendid.

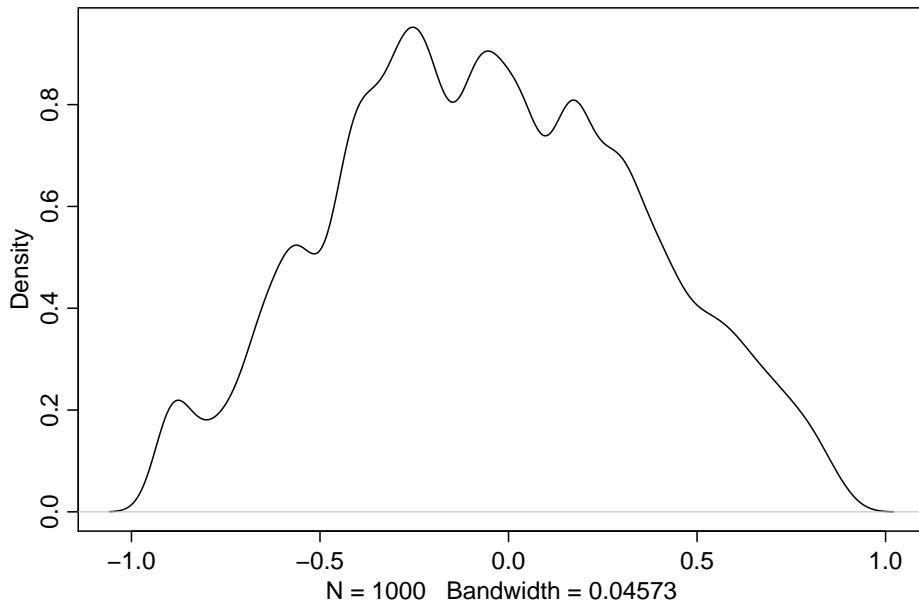


Figure 20.8: Posteeriор korrelatsioonile intercepti ja tõusu vahel.

tüdrukute-poiste erinevus üle kõikide koolide:

b_{sex1}

tüdrukute keskmise skoor üle kõikide koolide:

$\text{Intercept} + b_{\text{sex1}}$

ja poiste keskmise skoor üle kõikide koolide:

Intercept

Tõmbame mudelist ennustused 1., 2. ja 37. kooli poiste skooridele järgmisel semestril:

```
d.pred <- list(
  school = c(1, 2, 37),
  sex1 = 0
)

schools_sim <- rthinking::sim(schoolm2, data = d.pred)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
```

```
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
pred.p <- apply(schools_sim, 2, mean)
pred.p.PI <- apply(schools_sim, 2, PI)
```

NB! kasutades `rethinking::sim()` saame me enustused andmepunktide (üksikute poiste tasemel). Antud juhul jäab ennustuse kohaselt esimeses koolis 89% individuaalseid skoore vahemikku 61-132 punkti 200-st võimalikust.

Kui meid huvitab hoopis nende koolide keskmene skoor järgmisel semestril, siis kasuta `rethinking::sim()` asemel `rethinking::link()` funktsiooni.

```
schools_sim <- link(schools_m2, data = d.pred)
#> [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
pred.p <- apply(schools_sim, 2, mean)
pred.p.PI <- apply(schools_sim, 2, PI)
pred.p.PI
#> [,1] [,2] [,3]
#> 5% 32.6 34.4 44.3
#> 94% 46.2 39.9 49.7
```

Esimeses koolis jäab keskmene poiste skoor 89% tõenäosusega vahemikku 33 kuni 46 punkti.

```
compare(schools_m1, schools_m2)
#>           WAIC pWAIC dWAIC weight   SE   dSE
#> schools_m1 11734  55.8    0.0    0.7 57.2   NA
#> schools_m2 11736  60.8    1.7    0.3 57.2 1.38
```

Tundub, et tõusude vabakslaskmine oli hea mõte. Ma saan hästi pihta, et erinevad koolid õpetavad matemaatikat erineva kvaliteediga. Aga miks peaks erinevates Inglismaa koolides olema erinev vahe poiste ja tüdrukute matemaatikateadmistel? Kas olukorras kus meil on hea kool, läheb see vahe väiksemaks või suuremaks? Tehke kindlaks!!! võrrelge graafiku slope vs. intercept.

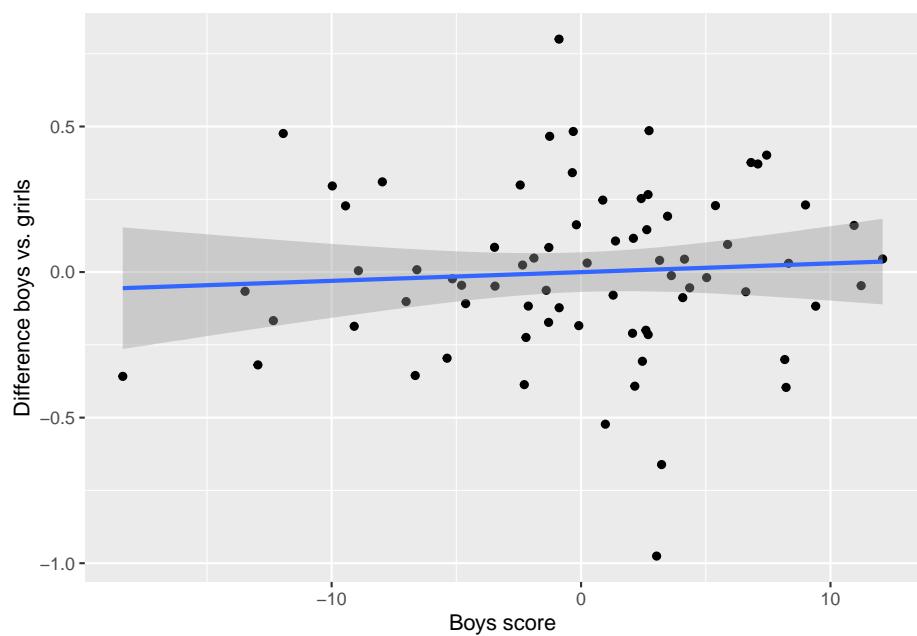


Figure 20.9: mida suurem on koolis poiste skoor, seda väiksem on poiste ja tüdrukute erinevus

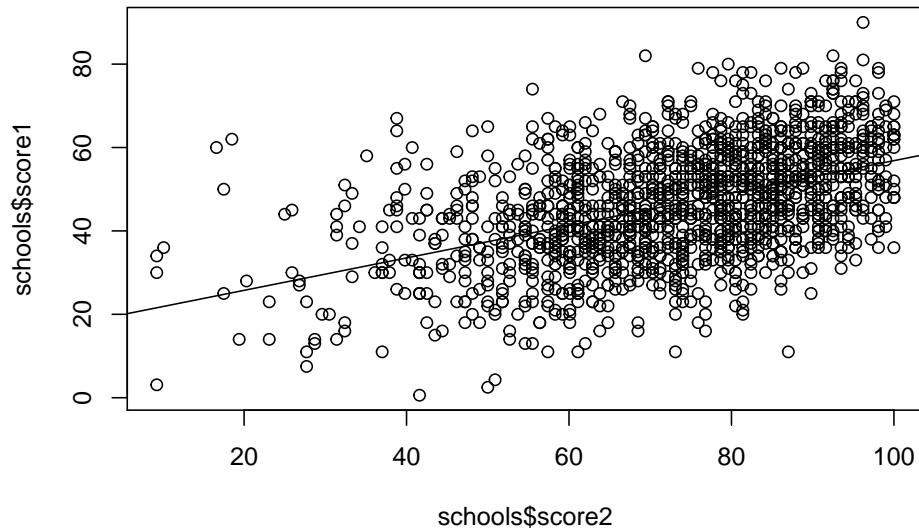


Figure 20.10: score1 vs. score2

Tõepoolest: mida suurem on koolis poiste skoor (parem kool), seda väiksem on poiste ja tüdrukute erinevus. Aga seos on kaunis nõrk!

Muidel sel joonisel tähendavad negatiivsed väärtsused alla keskmist väärust, mitte tingimata negatiivset erinevust või negatiivset skoori. Miks?

Arvutage nüütid poiste ja tüdrukute keskmise skoor kooli kaupa ja vaadake uuesti sõltuvust samasse erinevusesse. Mis on õigem viis: kas fittida ilma interceptita mudel (nagu eelmises peatükis) ja kasutada otse selle koefitsiente või kasutada meie m2 mudelit ning arvutada selle mudeli koefitsientide põhjal uus statistik (kaalutud keskmise näiteks)? Miks?

20.8 Hierarhiline mudel pidevate prediktoritega

Sin püüame ennustada score1 mõju score2 väärtsusele.

```
plot(schools$score2, schools$$score1)
abline(lm(score1 ~ score2, data = schools))
```

Kõigepealt lihtne regressioon `lm()` funktsiooniga (see ei ole hierarhiline mudel).

```
lm(score1 ~ score2, data = schools)
#>
#> Call:
#> lm(formula = score1 ~ score2, data = schools)
```

```
#>
#> Coefficients:
#> (Intercept)      score2
#>     17.971        0.389
```

score2 tõus 1 punkti võrra tõstab score1-e 0.39 punkti võrra.

Modelleerime seost üle Bayesi hierarhilise mudeli, kus ainult Intercept on vabaks lastud.

```
glimmer(score1 ~ score2 + (1 | school), data = schools)
#> alist(
#>   score1 ~ dnorm(mu, sigma),
#>   mu <- Intercept +
#>     b_score2 * score2 +
#>     v_Intercept[school],
#>   Intercept ~ dnorm(0, 10),
#>   b_score2 ~ dnorm(0, 10),
#>   v_Intercept[school] ~ dnorm(0, sigma_school),
#>   sigma_school ~ dcauchy(0, 2),
#>   sigma ~ dcauchy(0, 2)
#> )

schoolm7 <- map2stan(alist(
  score1 ~ dnorm(mu, sigma),
  mu <- Intercept +
    b_score2 * score2 +
    v_Intercept[school],
  Intercept ~ dnorm(50, 50),
  b_score2 ~ dnorm(0, 10),
  v_Intercept[school] ~ dnorm(0, sigma_school),
  sigma_school ~ dcauchy(0, 2),
  sigma ~ dcauchy(0, 2)
), data = schools)
```

Siin ei ole individuaalsed interceptid tõlgenduslikult informatiivsed, aga nende sissepanek parandab mudeli ennustust beta koefitsiendile (beta läheb väiksemaks ja ebakindlus selle hinnangu ümber kasvab).

```
precis(schoolm7, depth = 2)
```

Siin tuleb beta veidi väiksem - 0.36. Kuna sigma_school < sigma, siis tundub, et koolide vaheline varieeruvus on väiksem kui laste vaheline varieeruvus (sigma on üle kõigi koolide). iga kooli baastase tuleb Intercept + v_Intercept[] aga selle mudeli järgi on kõikide koolide score2 ja score1 sõltuvus sama tugevusega.

Laseme siis ka tõusud vabaks

```

glimmer(score1 ~ score2 + (1 + score2 | school), data = schools)
#> alist(
#>   score1 ~ dnorm(mu , sigma ),
#>   mu <- Intercept +
#>     b_score2*score2 +
#>     v_Intercept[school] +
#>     v_score2[school]*score2,
#>   Intercept ~ dnorm(0,10),
#>   b_score2 ~ dnorm(0,10),
#>   c(v_Intercept,v_score2)[school] ~ dmvnorm2(0,sigma_school,Rho_school),
#>   sigma_school ~ dcauchy(0,2),
#>   Rho_school ~ dlkjcorr(2),
#>   sigma ~ dcauchy(0,2)
#> )
schoolm5 <- map2stan(alist(
  score1 ~ dnorm(mu , sigma ),
  mu <- Intercept + b_score2 * score2 +
  v_Intercept[school] +
  v_score2[school] * score2,
  Intercept ~ dnorm(50, 25),
  b_score2 ~ dnorm(0, 10),
  c(v_Intercept, v_score2)[school] ~ dmvnorm2(0, sigma_school, Rho_school),
  sigma_school ~ dcauchy(0, 2),
  Rho_school ~ dlkjcorr(2),
  sigma ~ dcauchy(0, 2)
), data = schools)

```

nüüd saame igale koolile arvutada oma intercepti ja oma tõusu (ikka samamoodi: Intercept + v_intercept[] ja b_score2 + v_score2[])

```

precis(schoolm5, depth = 2)

schoolm6 <- map2stan(alist(
  score1 ~ dnorm(mu , sigma),
  mu <- Intercept + b_score2 * score2,
  Intercept ~ dnorm(50, 50),
  b_score2 ~ dnorm(0, 10),
  sigma ~ dcauchy(0, 2)
), data = schools)

```

m2 on selgelt parem mudel, kuigi m3 hinnangud interceptidele on suurema ebakindlusega. beta on nyyd 0.35

```

compare(schoolm7, schoolm6, schoolm5)
#>           WAIC pWAIC dWAIC weight    SE   dSE
#> schoolm5 11380  78.4    0.0      1 56.4   NA

```

```
#> schoolm7 11416  59.5  35.5      0 56.0 11.6
#> schoolm6 11862    3.3 482.0      0 54.6 41.6
```

0-mudel, mis on kõige kehvem, on kõige suurema betaga ja kõige väiksema ebakindlusega selle ümber. See on tavaline — hierarhiline mudel modelleerib ebakindlust paremini (realistlikumalt) ja vähendab üle-fittimise ohtu (beta tuleb selle võrra väiksem).

```
precis(schoolm7, depth = 2)
```


Chapter 21

brms

brms on pakett, mis võimaldab kirjutada lihtsat süntaksit kasutades ka üsna keerulisi mudeleid ja need Stan-is fittida. Brms on loodud Paul Bürkneri poolt (<https://github.com/paul-buerkner/brms>), mis on oma kasutuslihtsuse tõttu jõudnud isegi ette Stani meeskonna arendatavast analoogsest paketist **rstanarm** (<https://github.com/stan-dev/rstanarm/blob/master/README.md>). Paul Bürkner oli mõned aastad tagasi psühholoogia doktorant, kes kirjutas brms-i algselt naljaviluks oma doktoriprojekti kõrvalt, mille tulemusel on ta praegu teatud ringkonnis tuntum kui Lady Gaga.

rstanarm, mide me siin ei käsitle, püüab pakkuda tavalisete sageduslikele meetoditele (ANOVA, lineaarne regressioon jne) bayesi analooge, mille mudeli spetsifikatsioon ja väljund erineks võimalikult vähe tavalisest baas-R-i töövoost. **brms** on keskendunud mitmetasemelistele mudelitele ja kasutab põhimõtteliselt **lme4** (<https://github.com/lme4/lme4/>) mudelite keelt. Loomulikult saab **brms**-is fittida ka lineaarseid ja mitte-lineaareid ühetasemelisi mudeleid, nagu ka imputeerida andmeid ja teha palju muud (nagu näiteks pitsat valmistada).

```
library(tidyverse)
library(skimr)
library(brms)
library(loo)
library(broom)
library(bayesplot)
library(gridExtra)
library(mice)
library(pROC)
```

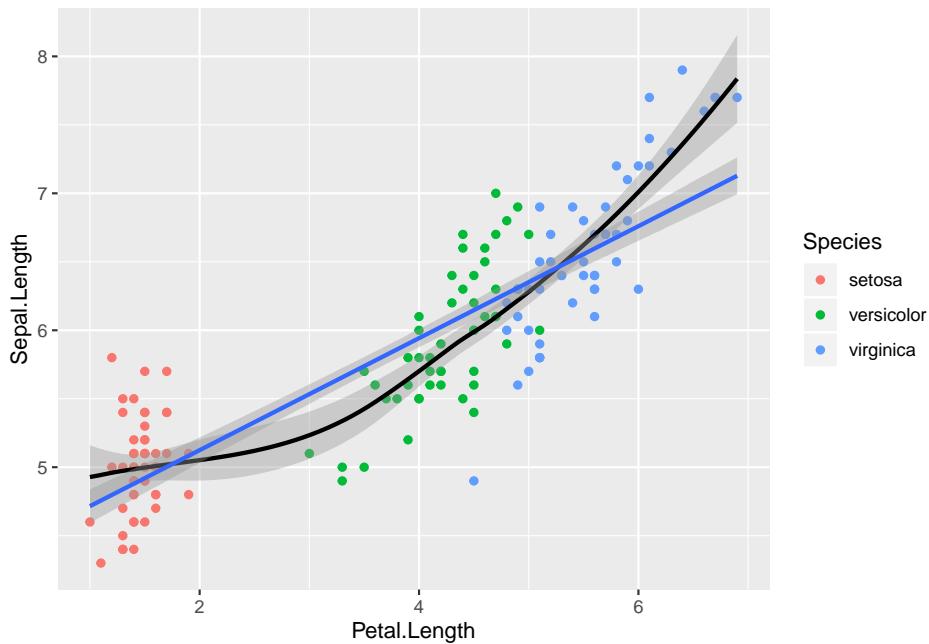
21.1 brms-i töövoog

brms-iga modelleerimisel on mõned asjad, mida tuleks teha sõltumata sellest, millist mudelit te parajasti fitite. Kõigepealt peaksite kontrollima, et mcmc ahelad on korralikult jooksnud (divergent transitions, rhat ja ahelate visuaalne inspekteerimine). Lisaks peaksite tegema posterioorse prediktiivse plotti ja vaatama, kui palju mudeli poolt genereeritud uued valimid meenutavad teie valimit. Samuti peaksite joonisel plottima residuaalid. Kui te inspekteerite fititud parametrite väärtsusi, siis tehke seda posteeriorite tasemel, k.a. koos veapiiridega. Kindlasti tuleks ka plottida mudeli ennustused koos usalduspiiridega.

Enne tõsist mudeldamist kiikame irise andmetabelisse

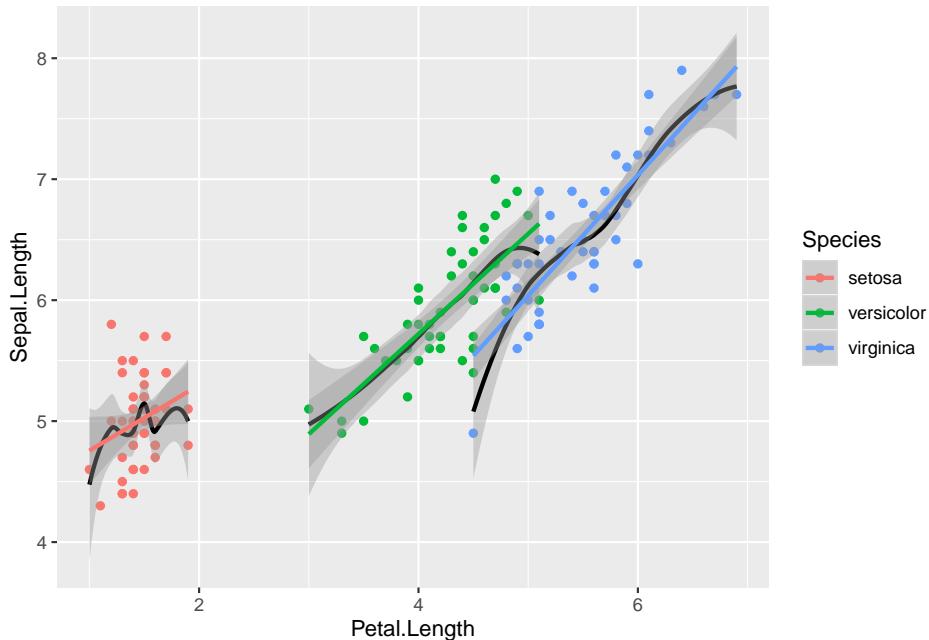
```
skim(iris)
#> Skim summary statistics
#> n obs: 150
#> n variables: 5
#>
#> -- Variable type:factor -----
#>   variable missing complete   n n_unique          top_counts
#>   Species        0     150 150            3 set: 50, ver: 50, vir: 50, NA: 0
#>   ordered        FALSE
#>
#> -- Variable type:numeric -----
#>   variable missing complete   n mean    sd   p0  p25  p50  p75  p100
#>   Petal.Length      0     150 150 3.76 1.77 1   1.6 4.35 5.1  6.9
#>   Petal.Width       0     150 150 1.2  0.76 0.1 0.3 1.3  1.8  2.5
#>   Sepal.Length      0     150 150 5.84 0.83 4.3 5.1 5.8  6.4  7.9
#>   Sepal.Width       0     150 150 3.06 0.44 2   2.8 3   3.3  4.4

ggplot(iris, aes(Petal.Length, Sepal.Length)) +
  geom_point(aes(color = Species)) +
  geom_smooth(method = "loess", color = "black") +
  geom_smooth(method = "lm")
```



Loess fit viitab, et 3 liiki ühe sirgega mudeldada pole võib-olla optimaalne lahendus.

```
ggplot(iris, aes(Petal.Length, Sepal.Length, color = Species)) +
  geom_point() +
  geom_smooth(method = "loess", aes(group = Species), color = "black") +
  geom_smooth(method = "lm")
```



Nüüd on loess ja lm heas kooskõlas - seos $y \sim x$ vahel oleks nagu enam-vähem lineaarne. Siit tuleb ka välja, et kolme mudeli tõusud on sarnased, interceptid erinevad.

21.1.1 Kiire töövoog

Minimaalses töövoos anname ette võimalikult vähe parameetreid ja töötame mudeliga nii vähe kui võimalik. See on mõeldud ülevaatena Bayesi mudeli fittimise põhilistest etappidest

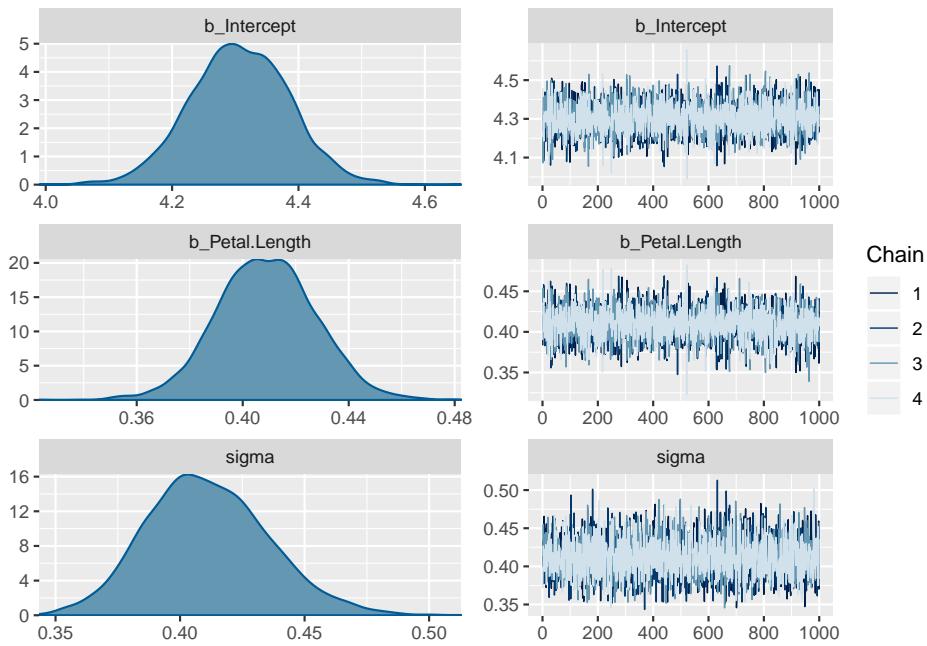
Mudeli fittimine:

```
m_kiire <- brm(Sepal.Length ~ Petal.Length, data = iris)
write_rds(m_kiire, path = "data/m_kiire.fit")
```

Priorid on brms-i poolt ette antud ja loomulikult ei sisalda mingit teaduslikku informatsiooni. Nad on siiski “nõrgalt informatiivsed” selles mõttes, et kasutavad parametriseringuid, mis enamasti võimaldavad mcmc ahelatel normaalset joosta. Järgmises ptk-s õpime ise prioreid määrama.

Posteriorid ja mcmc ahelate konvergents

```
plot(m_kiire)
```



Fiti kokkuvõte - koefitsiendid ja nende fittimise edukust hindavad statistikud (Eff.Sample, Rhat)

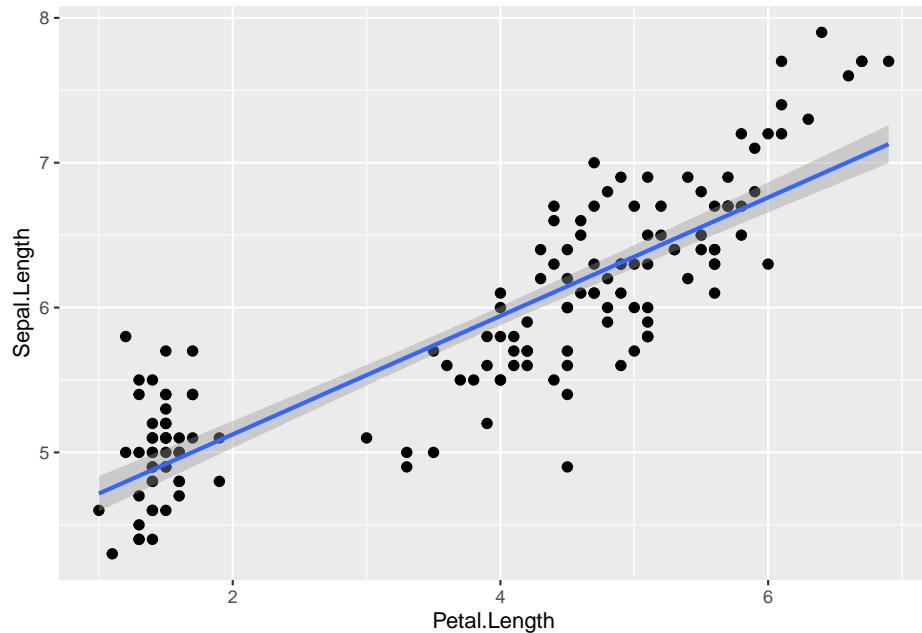
```
tidy(m_kiire)
#>   term estimate std.error lower upper
#> 1 b_Intercept 4.307    0.0780 4.179 4.434
#> 2 b_Petal.Length 0.409    0.0187 0.379 0.439
#> 3 sigma 0.411    0.0243 0.373 0.453
#> 4 lp__ -85.338   1.2173 -87.757 -84.015
```

Eff.Sample näitab efektiivset valimi suurust, mida ahelad on kasutanud. See on suht keeruline mõiste, aga piisab, kui aru saada, et see näitaja ei tohiks olla madalam kui paarkümmend.

Rhat on statistik, mis vaatab ahelate konvergentsi. Kui Rhat > 1.1, siis on kuri karjas. Rhat 1.0 ei tähenda paraku, et võiks rahulikult hingata – tegu on statistikuga, mida saab hästi tõlgendada häda kuulutajana, aga liiga sageli mitte vastupidi.

Ennustav plot ehk *marginal plot* – mudeli fit 95% CI-ga.

```
plot(marginal_effects(m_kiire), points = TRUE)
```



21.1.2 Põhjalikum töövoog

Põhiline erinevus eelmisega on suurem tähelepanu prioritele, mudeli fittimise diagnostikale ning tööl fititud mudeliga.

21.1.3 Spetsifitseerime mudeli, vaatame ja muudame vaiseprioreid

brms-i default priorid on konstrueeritud olema üsna väheinformatiivsed ja need tuleks enamasti informatiivsematega asendada. Igasse priorisse tuleks panna nii palju informatsiooni, kui teil on vastava parameetri kohta. Kui te mõne parameetri kohta ei oska öelda, millised oleks selle mõistlikud oodatavad väärustused, siis saab piirduda brms-i antud vaiseväärustustega. Samas, kui keerulisemad mudelid ei taha hästi joosta (mida tuleb ikka ette), siis aitab sageli priorite kitsamaks muutmine.

```
get_prior(Sepal.Length ~ Petal.Length + (1 | Species),
           data = iris)
#>          prior     class      coef   group resp dpar npar bound
#> 1          b
#> 2          b Petal.Length
#> 3 student_t(3, 6, 10) Intercept
#> 4 student_t(3, 0, 10)    sd
```

```
#> 5           sd      Species
#> 6           sd      Intercept Species
#> 7 student_t(3, 0, 10) sigma
```

Me fitime pedagoogilistel kaalutlustel shrinkage mudeli, mis tõmbab 3 liigi lõikepunkte natuke keskmise lõikepunkt suunas. On vaieldav, kas see on irise andmestiku juures mõistlik strateegia, aga me teeme seda siin ikkagi.

Mitmetasemeline shrinkage mudel on abinõu ülefittimise vastu. Mudelite võrdlemisel otsitakse kompromissi - ehk mudeli, mille ennustused oleks andmepunktidele võimalikult lähedal ilma, et see mudel oleks liiga keeruliseks aetud (keerulisus on proportsionaalne mudeli parameetrite arvuga).

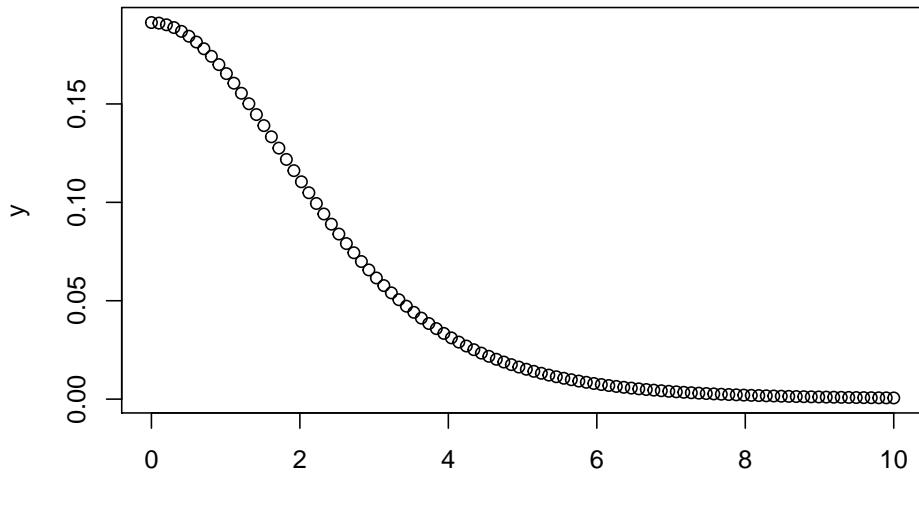
Prioreid muudame nii:

```
prior <- c(prior(normal(6, 3), class = "Intercept"),
            prior(normal(0, 1), class = "b"),
            prior(student_t(6, 0, 2), class = "sigma"))
```

Me valime siin nn väheinformariivsed priorid, nii et regressiooni tulemus on suht hästi võrreldav lme4 sagedusliku mudeliga. “b” koefitsiendi priorile (aga mitte “sigma” ega “Intercept”-le) võib anda ka ülemise ja/või alumise piiri `prior(normal(0, 1), class = "b", lb = -1, ub = 10)` ütleb, et “b” prior on nullist erinev ainult -1 ja 10 vahel. “sigma” priorid on automaatselt `lb = 0`-ga, sest varieeruvus ei tohi olla negatiivne.

Alati tasub prioreid pildil vaadata, et veenduda nende mõistlikuses.

```
x <- seq(0, 10, length.out = 100)
y <- dstudent_t(x, df = 6, mu = 0, sigma = 2, log = FALSE)
plot(y ~ x)
```



Sigma prior, mida **brms** kasutab, on vaikimisi pool sümmeetrilisest jaotusest, mis lõigatakse nulli kohalt pooleks nii, et seal puuduvad < 0 väärused (seega ei saa varieeruvuse posteerior minna alla nulli).

Me võime ka prioreid ilma likelihoodideta (tõepärafunktsioonideta) läbi mudeli lasta, misjärel tömbame fititud mudelist priorite valimid (neid võiks kutsuda ka "priorite posteerioriteks") ja plotime kõik priorid koos. Seda pilti saab siis võrrelda koos andmetega fititud mudeli posteerioritega. Selle võimaluse kasutamine on tõusuteel, sest keerulisemate mudelite puhul võib priorite ükshaaval plottimine osutuda eksitavaks.

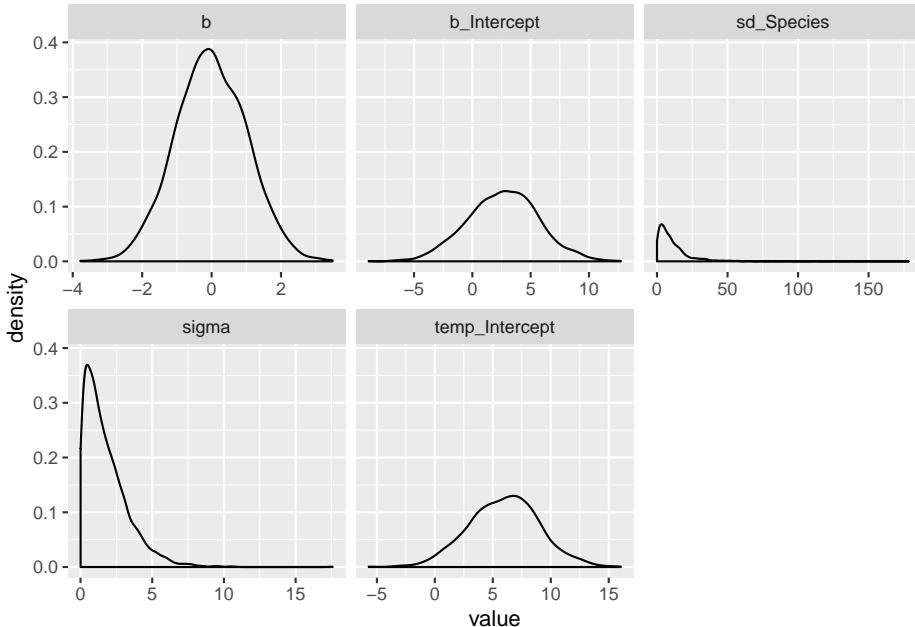
Tekitame priorite valimid, et näha oma priorite mõistlikust (`brm()` argument `on sample_prior = TRUE`). Ühtlasi fitime ka oma mudeli koos andmete ja prioritega.

```
m1 <- brm(Sepal.Length ~ Petal.Length + (1 | Species),
            data = iris,
            prior = prior,
            family = gaussian,
            warmup = 1000,
            iter = 2000,
            chains = 3,
            cores = 3,
            sample_prior = TRUE)
write_rds(m1, path = "data/m1.fit")
```

Me fittisime mudeli `m1` kaks korda: nii andmetega (selle juurde jõuame varsti), kui ka ilma andmeteta. Kui panna sisse `sample_prior = "only"`, siis jookseb mudel ilma andmeteta, ja selle võrra kiiremini. Vaikeväärtus on `sample_prior = "no"`, mis tähendab, et fititakse ainult üks mudel - koos andmetega. Ilma andmeteta (likelihoodita) fitist saame tömmata priorite mcmc valimid, mille ka

järgmiseks plotime.

```
prior_samples(m1) %>%
  gather() %>%
  ggplot() +
  geom_density(aes(value)) +
  facet_wrap(~ key, scales = "free_x")
```



Kui kasutame `sample_prior = "only"` variandi, siis on esimene koodirida erinev:

```
samples1 = as.data.frame(m1$fit).
```

brms-i Intercepti priorite spetsifitseerimisel tasub teada, et **brms** oma sisemuses tsentreerib kõik prediktorid nullile ($x - \text{mean}(x)$), ja teie poolt ette antud prior peaks vastama neile tsentreeritud prediktoritele, kus kõikide prediktorite keskväärtus on null. Põhjus on, et tsentreeritud parametriseeringuga mudelid jooksevad sageli paremini. Alternatiiv on kasutada mudeli tavapärase süntaksi $y \sim 1 + x$ (või ekvivalentelt $y \sim x$) asemel süntaksit $y \sim 0 + \text{intercept} + x$. Sellisel juhul saab anda priorid tsentreerimata prediktoritele. Lisaks on **brms** selle süntaksi puhul nõus “ b ”-le antud prioreid vaikimisi ka intercepti fittimisel kasutama.

21.1.4 `brm()` funktsiooni argumendid:

- `family` - tõepärafunktsiooni tüüp (modelleerib y muutuja jaotust e likelihoodi)

- warmup - mitu sammu mcmc ahel astub, enne kui ahelat salvestama hakatakse. Tavaliselt on 500-1000 sammu piisav, et tagada ahelate konvergents. Kui ei ole, tõstke 2000 sammuni.
- iter - ahelate sammude arv, mida salvestatakse peale warmup perioodi. Enamasti on 2000 piisav. Kui olete nõus piirduma posteriori keskväärtuse arvutamisega ja ei soovi täpseid usaldusintervalle, siis võib piisata ka 200 sammust.
- chains - mitu sõltumatut mcmc ahelat jooksutada. 3 on hea selleks, et näha kas ahelad konvergeeruvad. Kui mitte, tuleks lisada informatiivsemaid prioreid ja/või warmupi pikkust.
- cores - mitu teie arvuti tuuma ahelaid jooksutama panna.
- adapt_delta - mida suurem number ($\max = 1$), seda stabiilsemalt, ja aeglasmalt, ahelad jooksevad.
- thin - kui ahel on autokorreleeritud, st ahela eelmine samm suudab ennustada järgnevaid (see on paha), siis saab salvestada näit ahela iga 5. sammu ($\text{thin} = 5$). Aga siis tuleks ka sammude arvu 5 korda tõsta. Vaikeväärtus on $\text{thin} = 1$. Autokorrelatsiooni graafilist määramist näitame allpool

Järgmine funktsioon trükiib välja Stan'i koodi, mis spetsifitseerib mudeli, mida tegelikult Stanis fittima hakatakse. See on väga kasulik, aga ainult siis kui tahate õppida otse Stanis mudeleid kirjutama:

```
make_stancode(Sepal.Length ~ Petal.Length, data = iris, prior = prior)
```

21.1.5 Fitime mudeleid ja võrdleme fitte.

Eelmises mudelis (m1) ennustame muutuja Sepal.Length väärtsusi Petal.Length väärtsuste põhjal shrinkage mudelis, kus iga irise liik on oma grups.

Teine mudel, mis sisaldab veel üht ennustavat muutujat (Sepal.Width):

```
m2 <- brm(Sepal.Length ~ Petal.Length + Sepal.Width + (1 | Species),
           data = iris,
           prior = prior)
write_rds(m2, path = "data/m2.fit")
```

Kolmandaks ühetasemeline mudel, mis vaatab kolme irise liiki eraldi:

```
m3 <- brm(Sepal.Length ~ Sepal.Width + Petal.Length * Species,
           data = iris,
           prior = prior)
write_rds(m3, path = "data/m3.fit")
```

Ja lõpuks mudel, mis paneb kõik liigid ühte patta:

```
m4 <- brm(Sepal.Length ~ Petal.Length + Sepal.Width,
            data = iris,
            prior = prior)
write_rds(m4, path = "data/m4.fit")
```

Siin me võrdleme neid nelja mudelit. Väikseim looic (leave-one-out information criterion) võidakab. See on suhteline võrdlus – looic abs väärustus ei mängi mingit rolli.

```
loo(m1, m2, m3, m4)
#> Output of model 'm1':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate    SE
#> elpd_loo     -53.2  8.3
#> p_loo        4.8   0.7
#> looic       106.4 16.6
#> -----
#> Monte Carlo SE of elpd_loo is 0.1.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
#>
#> Output of model 'm2':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate    SE
#> elpd_loo     -40.8  8.1
#> p_loo        5.6   0.7
#> looic       81.6 16.1
#> -----
#> Monte Carlo SE of elpd_loo is 0.1.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
#>
#> Output of model 'm3':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate    SE
#> elpd_loo     -40.0  7.9
#> p_loo        7.1   0.9
#> looic       80.0 15.8
```

```

#> -----
#> Monte Carlo SE of elpd_loo is 0.1.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
#>
#> Output of model 'm4':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate    SE
#> elpd_loo     -50.3   8.2
#> p_loo        3.5    0.5
#> looic       100.5  16.4
#> -----
#> Monte Carlo SE of elpd_loo is 0.0.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
#>
#> Model comparisons:
#>   elpd_diff se_diff
#> m3    0.0      0.0
#> m2   -0.8      1.7
#> m4  -10.3      5.0
#> m1  -13.2      5.3

```

Sin on m1 ja m2/m3 mudeli erinevus 25 ühikut ja selle erinevuse standardviga on 10 ühikut. 2 SE-d annab umbkaudu 95% usaldusintervalli, ja see ei kata antud juhul nulli. Seega järeldamene, et m2 ja m3, mis kasutavad ennustamiseks lisamuutujat, on selgelt eelistatud. Samas ei saa me öelda, et hierarhiline mudel m2 oleks parem või halvem kui interaktsioonimudel m3. Ka puudub oluline erinevus m1 ja m4 fiti vahel. Tundub, et selle ennustusjõu, mille me võidame lisaparametrit mudeldades, kaotame omakorda liike ühte patta pannes (neid mitte osaliselt iseseisvana modelleerides).

Alternatiivina kasutame `brms::waic` kriteeriumit mudelite võrdlemiseks. See töötab kiiremini kui LOO ja tõlgendus on sarnane - väikseim waic võidak jaabsolutväärtsi ei saa ükshaaval tõlgendada.

```

waic(m1, m2, m3, m4)
#> Output of model 'm1':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate    SE

```

```

#> elpd_waic    -53.2  8.3
#> p_waic       4.8   0.7
#> waic         106.4 16.6
#> Warning: 1 (0.7%) p_waic estimates greater than 0.4. We recommend trying
#> loo instead.
#>
#> Output of model 'm2':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate   SE
#> elpd_waic    -40.8  8.0
#> p_waic       5.6   0.7
#> waic         81.6 16.1
#>
#> Output of model 'm3':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate   SE
#> elpd_waic    -40.0  7.9
#> p_waic       7.0   0.9
#> waic         80.0 15.8
#> Warning: 1 (0.7%) p_waic estimates greater than 0.4. We recommend trying
#> loo instead.
#>
#> Output of model 'm4':
#>
#> Computed from 3000 by 150 log-likelihood matrix
#>
#>           Estimate   SE
#> elpd_waic    -50.3  8.2
#> p_waic       3.5   0.5
#> waic         100.5 16.4
#>
#> Model comparisons:
#>   elpd_diff se_diff
#> m3  0.0      0.0
#> m2 -0.8      1.7
#> m4 -10.3     5.0
#> m1 -13.2     5.3

```

Nagu näha, annavad LOO ja waic sageli väga sarnaseid tulemusi.

Me ei süvene LOOIC ega waic-i statistilisse mõttesse, sest bayesi mudelite võrdlemine on kiiresti arenev ala, kus ühte parimat lahendust pole veel leitud.

21.1.6 Vaatame mudelite kokkuvõtet

Lihitne tabel mudeli m2 fititud koefitsientidest koos 95% usalduspiiridega

<code>tidy(m2)</code>		term	estimate	std.error	lower	upper
#>						
#> 1		b_Intercept	1.710	1.0270	0.095	3.345
#> 2		b_Petal.Length	0.759	0.0645	0.651	0.866
#> 3		b_Sepal.Width	0.440	0.0839	0.303	0.576
#> 4		sd_Species_Intercept	1.725	1.5315	0.433	5.024
#> 5		sigma	0.313	0.0186	0.284	0.345
#> 6	r_Species[setosa, Intercept]		0.676	0.9925	-0.901	2.250
#> 7	r_Species[versicolor, Intercept]		-0.226	0.9842	-1.853	1.253
#> 8	r_Species[virginica, Intercept]		-0.642	0.9907	-2.277	0.809
#> 9		lp__	-50.433	2.3680	-54.788	-47.169

r_ prefiks tähendab, et antud koefitsient kuulub mudeli esimesele (madalamale) tasemele (Liigi tase) r - random - tähendab, et iga gruupi (liigi) sees arvutatakse oma fit. b_ tähendab mudeli 2. taset (keskmistatud üle köikide gruppide). 2. tasmel on meil intercept, b1 ja b2 töösud ning standardhälve y muutuja ennustatud andempunktide tasemel. 1. tasemel on meil 3 liigi interceptide erinevus üldisest b_Intercepti väärustest. Seega, selleks, et saada setosa liigi intercepti, peame tegema tehte $1.616 + 0.765$.

`tidy` funktsiooni tööd saab kontrollida järgmiste parameetrite abil:

```
tidy(x, parameters = NA, par_type = c("all",
  "non-varying", "varying", "hierarchical"), robust = FALSE,
  intervals = TRUE, prob = 0.9, ...)
```

`par_type = "hierarchical"` kuub grupi taseme parameetrite sd-d ja korrelatsioonid. `"varying"` kuub grupi taseme interceptid ja töösud (siis kui neid mudel-dadakse). `"non-varying"` kuub kõrgema taseme (grupi-ülesed) parameetrid. `robust = TRUE` annab estimate posteeriori mediaanina (vaikeväärustus FALSE annab selle aritmeetilise keskmisena posteeriorist).

Nüüd põhjalikum mudeli kokkuvõte:

```
m2
#> Warning: There were 17 divergent transitions after warmup. Increasing adapt_delta at
#> See http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
#> Family: gaussian
#>   Links: mu = identity; sigma = identity
#> Formula: Sepal.Length ~ Petal.Length + Sepal.Width + (1 | Species)
#>   Data: iris (Number of observations: 150)
#>   Samples: 3 chains, each with iter = 2000; warmup = 1000; thin = 1;
#>             total post-warmup samples = 3000
#>
```

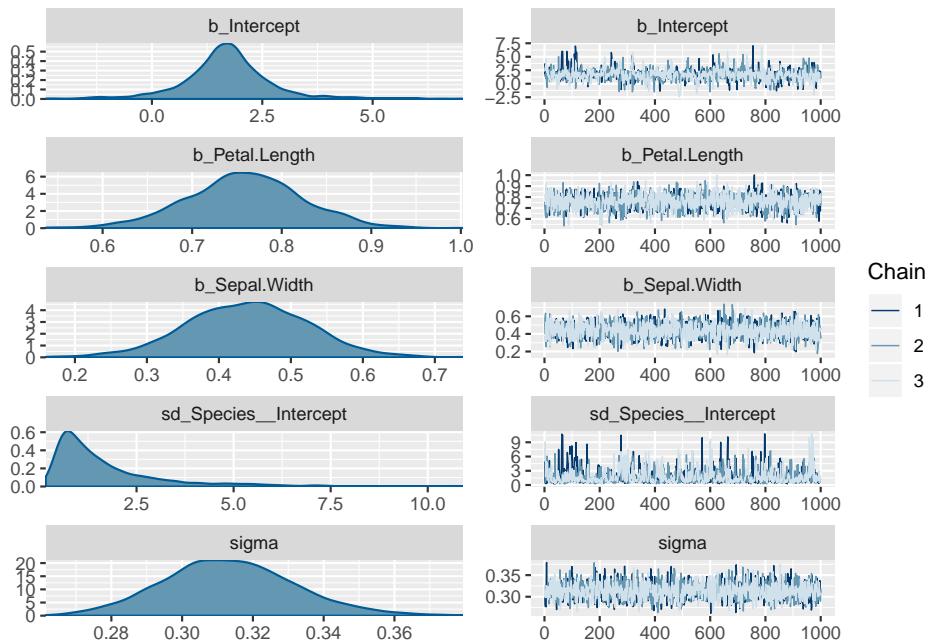
```
#> Group-Level Effects:
#> ~Species (Number of levels: 3)
#>           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
#> sd(Intercept)    1.72      1.53     0.36     5.98 1.00      591      813
#>
#> Population-Level Effects:
#>           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
#> Intercept       1.71      1.03    -0.36     4.08 1.01      725      678
#> Petal.Length    0.76      0.06     0.63     0.88 1.00     1524     1640
#> Sepal.Width     0.44      0.08     0.27     0.60 1.00     1937     1686
#>
#> Family Specific Parameters:
#>           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
#> sigma            0.31      0.02     0.28     0.35 1.00     1896     1525
#>
#> Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
#> is a crude measure of effective sample size, and Rhat is the potential
#> scale reduction factor on split chains (at convergence, Rhat = 1).
```

Siin on eraldi toodud grupi tasemel ja populatsiooni tasemel koefitsiendid ja gruppide vaheline sd ($= 1.72$). Pane tähele, et üldine varieeruvus σ $= 0.31$ on palju väiksem kui gruppide vaheline varieeruvus $sd(\text{Intercept}) = 1.72$. Seega on grupid üksteisest tugevalt erinevad ja neid tuleks võib-olla töesti eraldi modelleerida.

Divergentsed transitsioonid on halvad asjad - ahelad on läinud 17 korda metsa. Viisakas oleks adapt deltat tõsta või kitsamad priorid panna, aga 17 halba andmepunkti paarist tuhandest, mille mcmc ahelad meile tekitasid, pole ka mingi maailmalõpp. Nii et las praegu jäääb nagu on. Need divergentsed transitsioonid on kerged tekkima just mitmetasemelistes mudelites.

21.1.7 Plotime posteeriorid ja ahelad

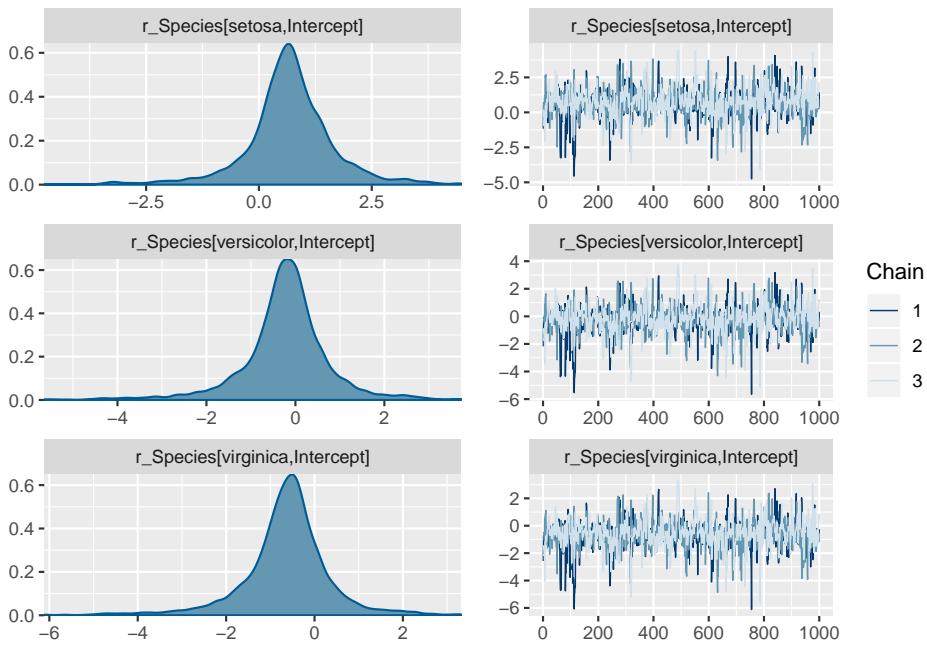
```
plot(m2)
```



Sit on näha, et ahelad on ilusti konvergeerunud. Ühtlasi on pildil posterioorsed jaotused fititud koefitsientidele.

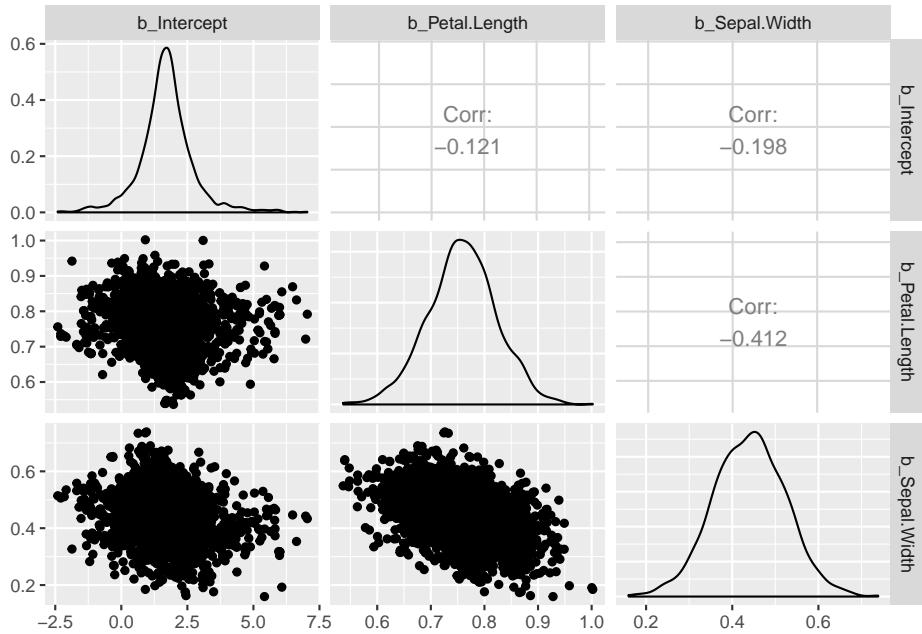
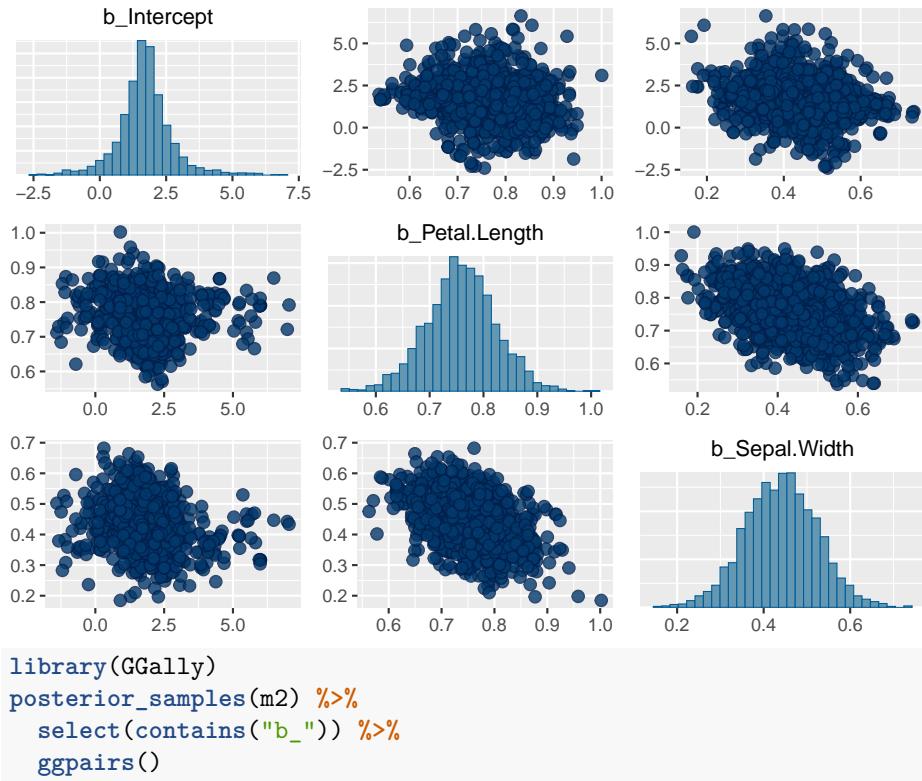
Regular expressioni abil saab plottida mudeli madalama taseme ahelaid & positioreid, mida `plot()` vaikimisi ei näita.

```
plot(m2, pars = "r_")
```



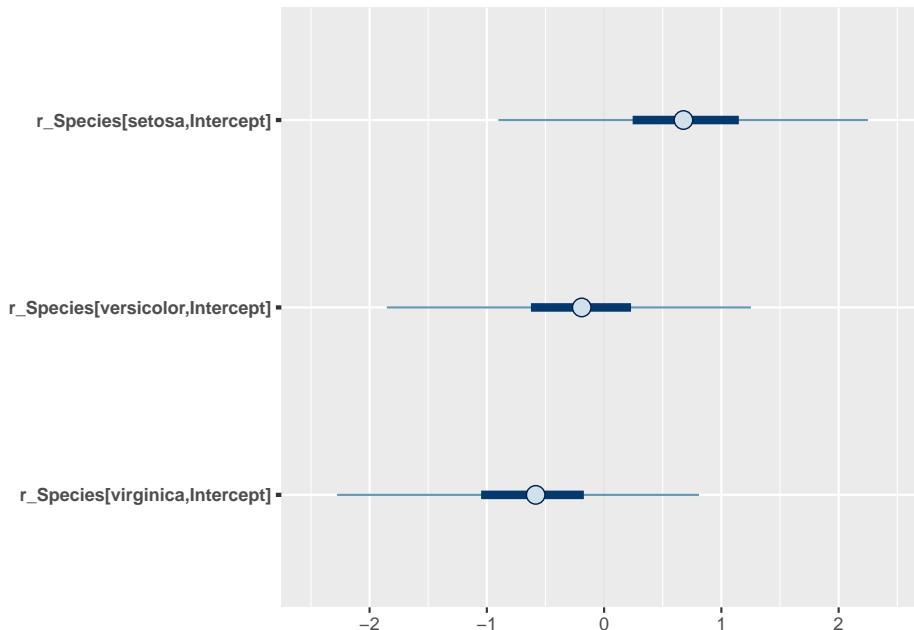
Vaatame korrelatsioone erinevate parameetrite posterioorsete valimite vahel. (Markovi ahelad jooksevad n-mõõtmelises ruumis, kus n on mudeli parameetrite arv, mille väärtsusi hinnatakse.) Pairs(m3) teeb pildi ära, aga ilusama pildi saab GGally::ggpairs() abil.

```
pairs(m2, pars = "b_")
```



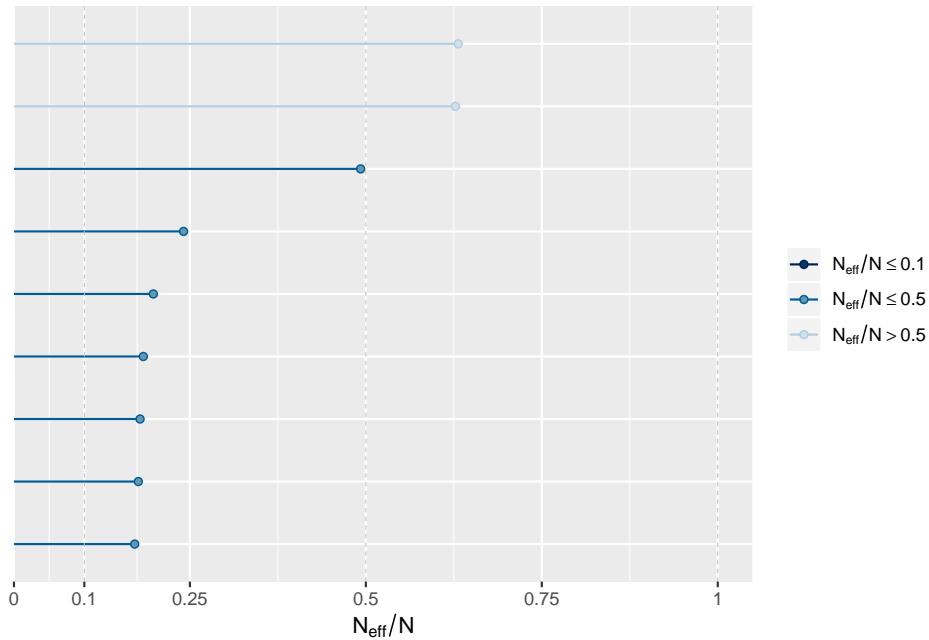
Siin on posteeriorite põhjal arvutatud 50% ja 95% CI ja see plotitud.

```
stanplot(m2, pars = "r_",
         type = "intervals")
```



`type` = argument sisestamine võimaldab plottida erinevaid diagnostilisi näitajaid. Lubatud sisendid on “hist”, “dens”, “hist_by_chain”, “dens_overlay”, “violin”, “intervals”, “areas”, “acf”, “acf_bar”, “trace”, “trace_highlight”, “scatter”, “rhat”, “rhat_hist”, “neff”, “neff_hist”, “nuts_acceptance”, “nuts_divergence”, “nuts_stepsize”, “nuts_treedepth” ja “nuts_energy”.

```
stanplot(m2, type = "neff")
```



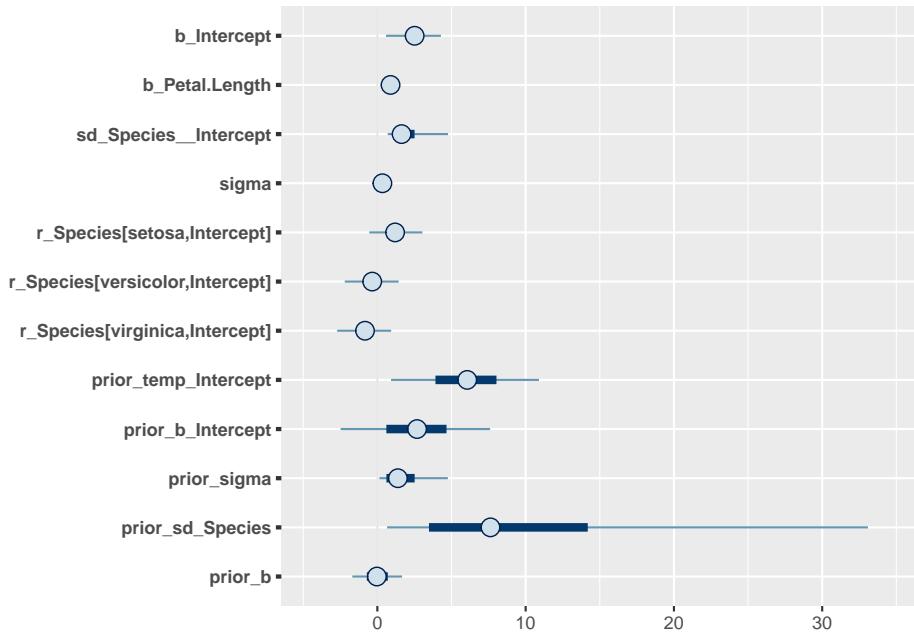
N_{eff} on efektiivne valimi suurus ja senikaua kuni N_{eff}/N suhe ei ole < 0.1 , pole põhjust selle pärast muretseda.

21.1.8 Korjame ahelad andmeraami ja plotime fititud koefitsiendid CI-dega

```
model <- posterior_samples(m1)
```

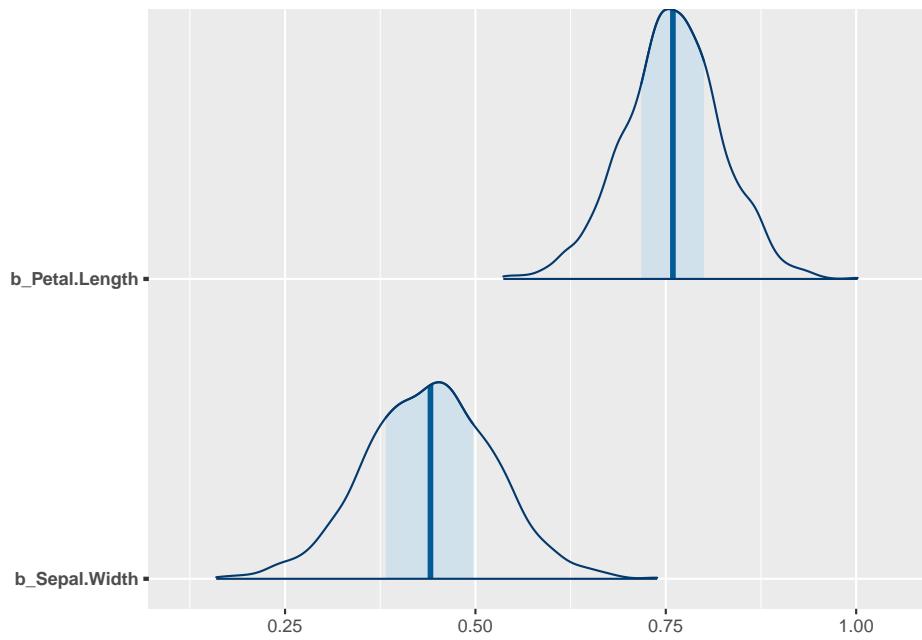
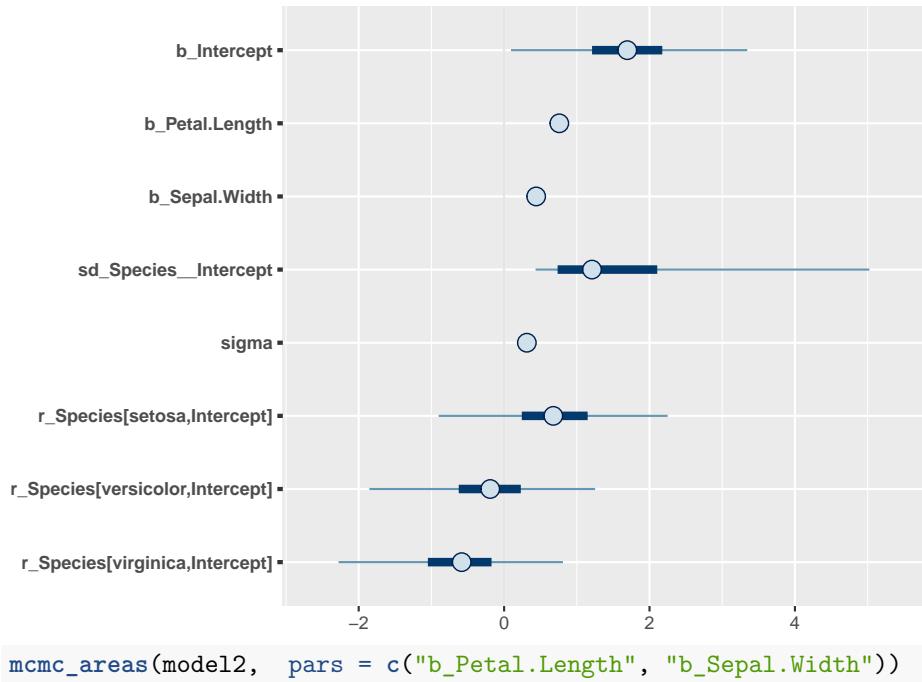
`mcmc_intervals()` on bayesplot paketi funktsioon. Me plotime 50% ja 95% CI-d.

```
pars <- colnames(model)
mcmc_intervals(model, regex_pars = "[^lp_]").
```



Näeme, et sigma hinnang on väga usaldusväärne, samas kui gruppide vahelise sd hinnang ei ole seda mitte (pane tähele posterioorse jaotuse ebasümmeetrilisust).

```
model2 <- posterior_samples(m2)
pars <- colnames(model2)
mcmc_intervals(model2, regex_pars = "[^lp__]")
```



21.1.9 Bayesi versioon R-ruudust

Kui suurt osa koguvarieeruvusest suudavad mudeli prediktorid seletada?

```
bayes_R2(m2)
#>   Estimate Est.Error Q2.5 Q97.5
#> R2      0.86   0.00831 0.84 0.873

bayes_R2(m1)
#>   Estimate Est.Error Q2.5 Q97.5
#> R2      0.833   0.0109 0.807 0.85
```

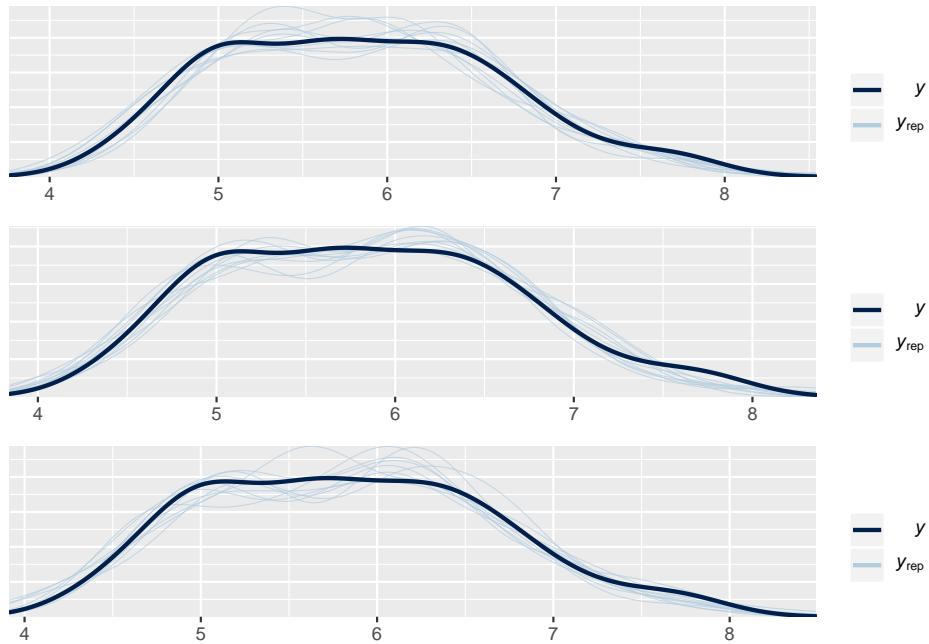
https://github.com/jgabry/bayes_R2/blob/master/bayes_R2.pdf Annab põhjenduse sellele statistikule (mille arvutamine erineb tavalisest vähimruutudega arvutatud mudeli R^2 -st).

21.1.10 Plotime mudeli poolt ennustatud valimeid – posterior predictive check

Kui mudel suudab genereerida simuleeritud valimeid, mis ei erine väga palju empiirilisest valimist, mille põhjal see mudel fititi, siis võib-olla ei ole see täiesti ebaõnnestunud mudeldamine. See on loogika posterioorse ennustava plati taga.

Vaatame siin simultaanselt kõigi kolme eelnevalt fititud mudeli simuleeritud valimeid (`y_rep`) võrdluses algsete andmetega (`y`):

```
purrr::map(list(m1, m2, m3), pp_check, nsamples = 10) %>%
  grid.arrange(grobs = ., nrow = 3)
```

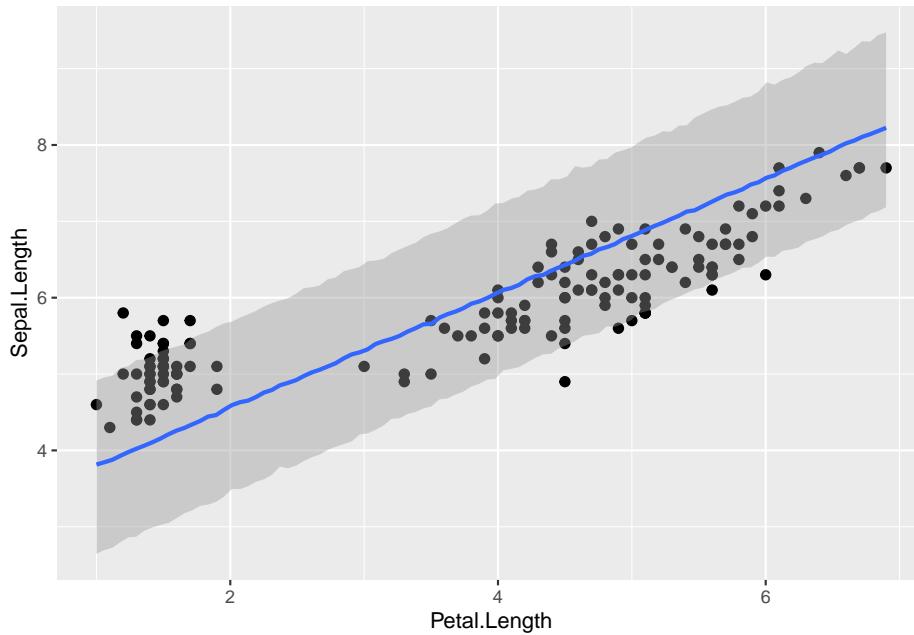


- y - tihedusplot empiirilistest andmetest
- y_{rep} – plotid mudeli poolt ennustatud iseseisvatest valimitest (igaüks sama suur kui empiiriline valim y) Jooniselt on näha, et m_3 ennustused on võrreldes m_1 ja m_2 -ga kõigemal kaugemal tegelikust valimist.

21.1.11 Plotime mudeli ennustusi - marginal effects plots

Teeme ennustused. Kõigepealt ennustame ühe keskmise mudeliga, mis ei arvesta mitmetasemelise mudeli madalamte tasemete koefitsientidega.

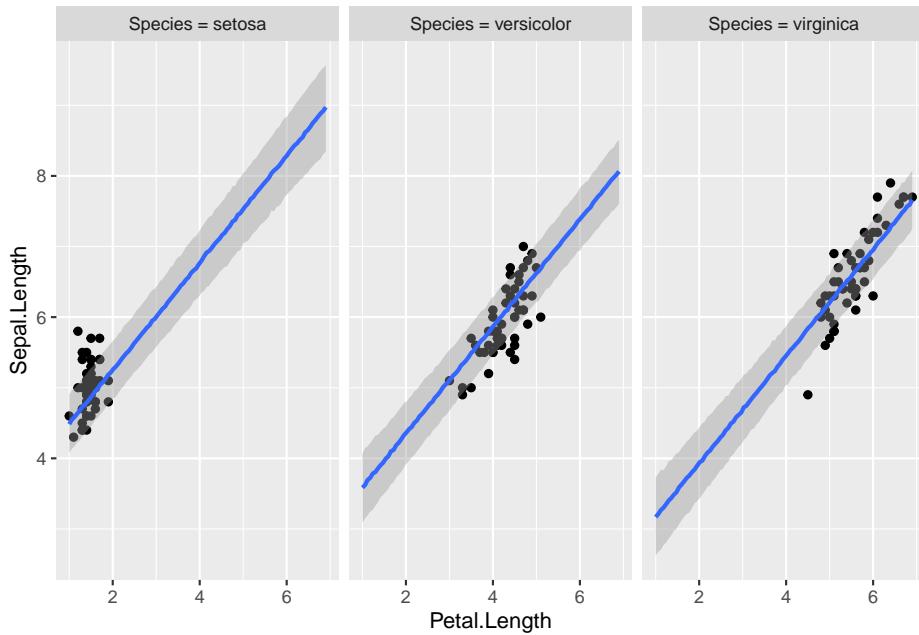
```
plot(marginal_effects(m2, effects = "Petal.Length", method = "predict", probs = c(0.1,
```



Ennustus on selles mõttes ok, et vaid väike osa punkte jäääb sellest välja, aga laiavõitu teine!

Nüüd ennustame sama mudeli põhjal igale liigile eraldi. Seega kasutame mudeli madalama taseme koefitsiente. Peame andma lisaparameetri `re_formula = NULL`, mis tagab, et ennustuse tegemisel kasutatakse ka mudeli madalama taseme koefitsiente.

```
plot(marginal_effects(m2, effects = "Petal.Length", method = "predict", conditions = make_conditions(m2)))
```

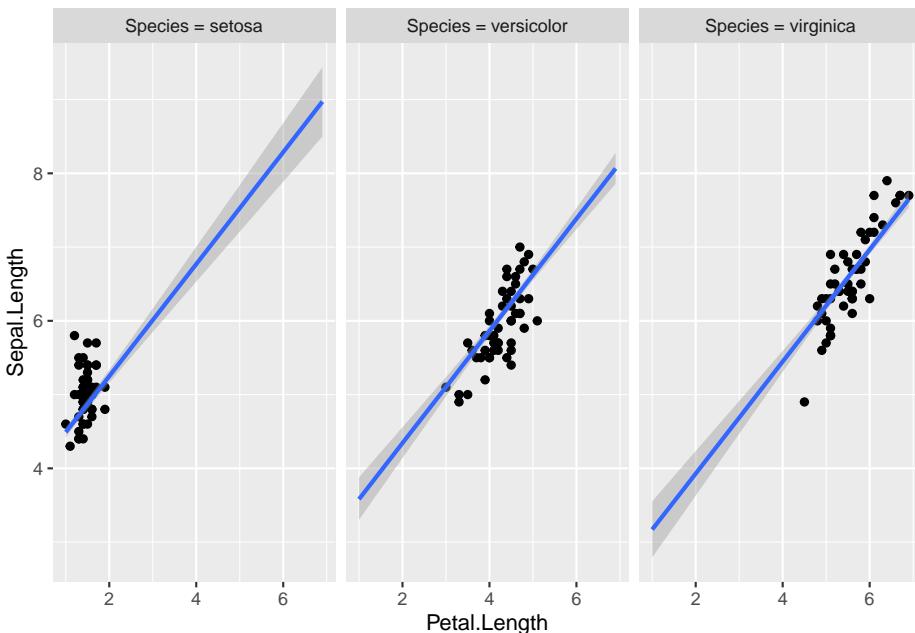


`method = "predict"` ennustab, millisesse vahemikku peaks mudeli järgi jäma 90% andmepunkte (k.a. uued andmepunktid, mida pole veel valimisse korjatud).

Tõesti, valdav enamus valimi punkte on intervallis sees, mis viitab et mudel töötab hästi. Seal, kus on rohkem punkte, on intervall kitsam ja mudel usaldusväärsem.

Järgneval pildil on `method = "fitted"`. Nüüd on enamus punkte väljaspool usaldusintervalle, mis sellel pildil mõõdavad meie usaldust regressioonijoone vastu.

```
plot(marginal_effects(m2, effects = "Petal.Length", method = "fitted", conditions = ma
```



```
method = "fitted" annab CI regressioonjoonele.
```

Argumendid:

- method – predict annab veapiirid (95% CI) mudeli ennustustustele andmepunkti tasemel. fitted annab veapiirid mudeli fitile endale (joonele, mis tähistab keskmist või kõige tõenäolisemat y muutuja väärust iga x-i väärusel)
- conditions - andmeraam, kus on kirjas mudeli nendele ennustavatele (x) muutujatele omistatud väärused, mida ei joonistata x teljele. Kuna meil on selleks mudeli madalamana taseme muutuja Species, siis on lisaks vaja määräta argument `re_formula = NULL`, mis tagab, et ennustuste tegemisel kasutatakse mudeli kõikide tasemetate fititud koefitsiente. `re_formula = NA` annab seestavu keskmise fiti üle kõigi gruppide (iri liikide)
- probs annab usaldusintervalli piirid.

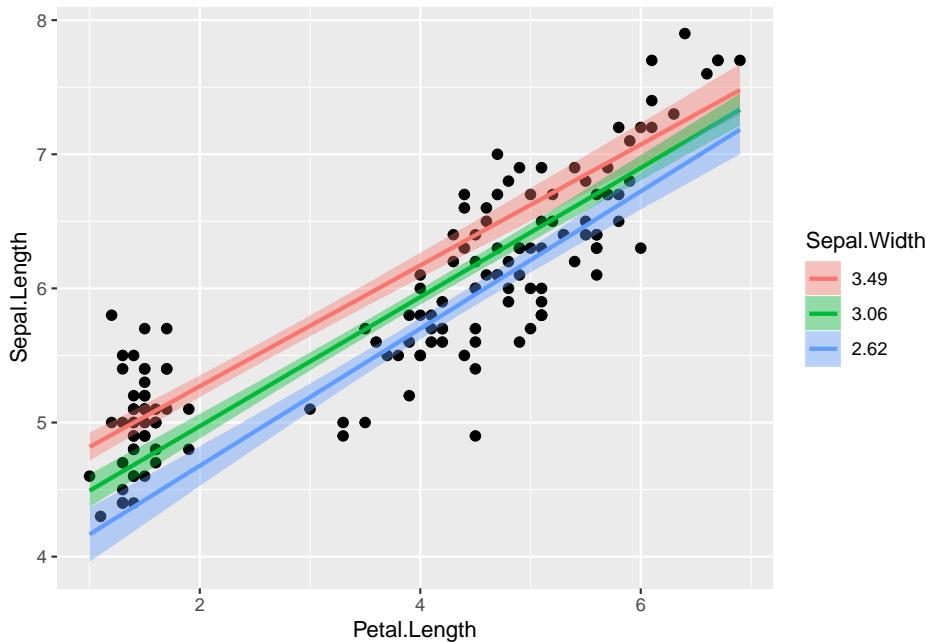
Pane tähele, et argument `points` (ja muud lisaargumendid, nagu näiteks `theme`) kuuluvald `plot()`, mitte `marginal_effects()` funktsioonile.

Tavaline interaktsionimudel, aga pidevatele muutujatele.

```
m5 <- brm(Sepal.Length ~ Petal.Length + Sepal.Width + Petal.Length * Sepal.Width,
            data = iris,
            prior = prior,
            family = gaussian)
write_rds(m5, path = "data/m5.fit")
```

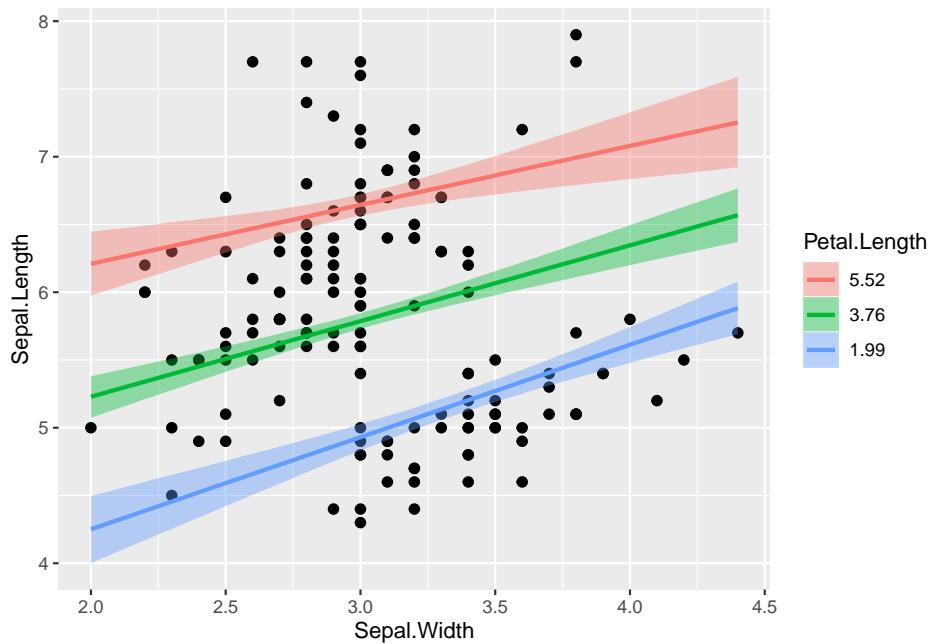
Kõigepealt plotime mudeli ennustused, kuidas Sepal Length sõltub Petal Lengthist kolmel erineval Sepal width väärtsel.

```
plot(marginal_effects(m5, effects = "Petal.Length:Sepal.Width"), points = TRUE)
```



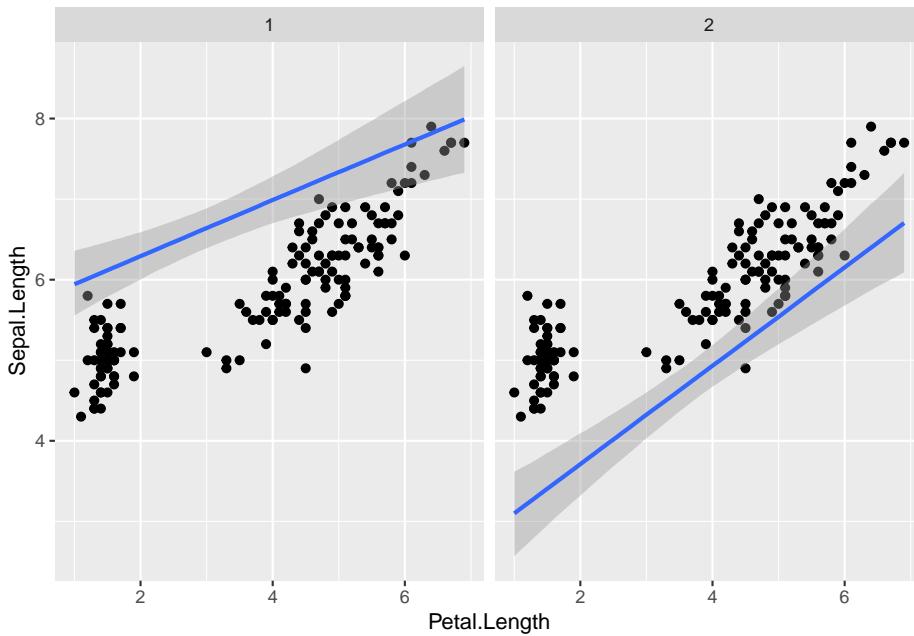
Ja siis sümmeetriliselt vastupidi.

```
plot(marginal_effects(m5, effects = "Sepal.Width:Petal.Length"), points = TRUE)
```



Siin lisame enda soovitud Sepal Width väärtsused (5 ja 1.2), mis on väljaspool seda, mida loodus pakub. Pane tähele ennustuse laiemaid CI-e.

```
conditions <- data.frame(Sepal.Width = c(5, 1.2))
plot(marginal_effects(m5, effects = "Petal.Length", conditions = conditions, re_formula = NULL),
```



21.1.12 Alternatiivne tee

Alternatiivne millele? Teeme tabeli nende väärustega, millele tahame mudeli ennustusi. Tabelis newx on spetsifitseeritud mudeli kõikide X muutujate väärustused! Me ennustame Y väärustusi paljudel meie poolt võrdse vahemaaga ette antud petal length väärustustel, kusjuures me hoiame sepal width väärtsuse alati konstantsena tema valimi keskmisel väärtsusel ja vaatame ennustusi eraldi kahele liigile kolmest. Liigid on mudeli madala taseme osad, seega kasutame ennustuste tegemisel mudeli kõikide tasemete koefitsiente.

```
newx <- expand.grid(Petal.Length = modelr::seq_range(iris$Petal.Length, n = 150),
                      Sepal.Width = mean(iris$Sepal.Width),
                      Species = c("setosa", "virginica"))
```

`expand.grid()` lõõb tabeli pikaks nii, et kõik võimalikud kombinatsioonid 3st muutujast on täidetud väärustega.

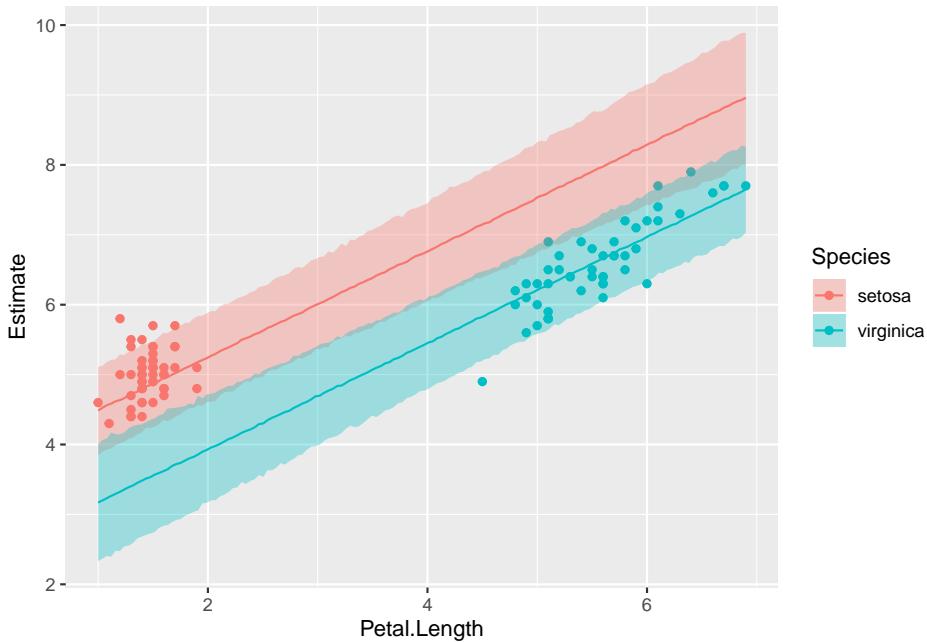
`re_formula` NULL mudeldab eraldi liigid eraldi mudeli madalama taseme (liikide sees) koefitsiente kasutades

```
predict_interval_brms2 <- predict(m2, newdata = newx, re_formula = NULL) %>%
  cbind(newx, .)
head(predict_interval_brms2)
#>   Petal.Length Sepal.Width Species Estimate Est.Error Q2.5 Q97.5
#> 1      1.00     3.06  setosa    4.49   0.325 3.85  5.11
#> 2      1.04     3.06  setosa    4.53   0.314 3.89  5.12
#> 3      1.08     3.06  setosa    4.56   0.319 3.93  5.18
#> 4      1.12     3.06  setosa    4.58   0.328 3.94  5.21
#> 5      1.16     3.06  setosa    4.62   0.315 3.99  5.22
#> 6      1.20     3.06  setosa    4.63   0.319 4.01  5.29
```

`predict()` ennustab uusi petal length väärtsusi (Estimate veerg) koos usaldusinetralliga neile väärustustele

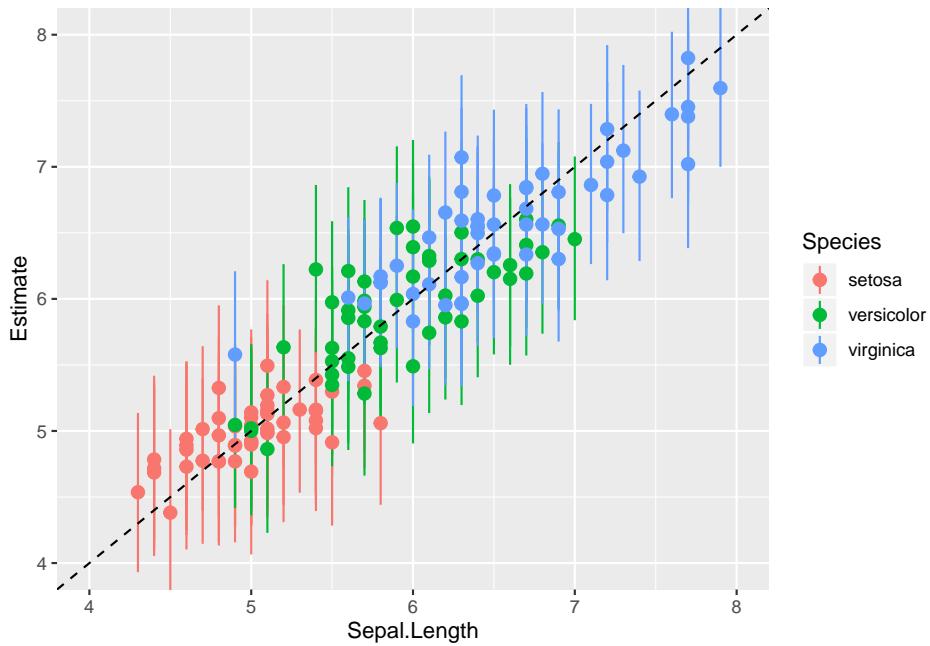
Siin siis eraldi ennustused kahele liigile kolmest, kaasa arvatud petal length väärtsusvahemikule, kus selle liigi isendeid valimis ei ole (ja võib-olla ei saagi olla)

```
no_versicolor <- filter(iris, Species != "versicolor")
ggplot(data = predict_interval_brms2, aes(x = Petal.Length, y = Estimate)) +
  geom_point(data = no_versicolor, aes(Petal.Length, Sepal.Length, color = Species)) +
  geom_line(aes(color = Species)) +
  geom_ribbon(aes(ymin = Q2.5, ymax = Q97.5, fill = Species), alpha = 1/3)
```



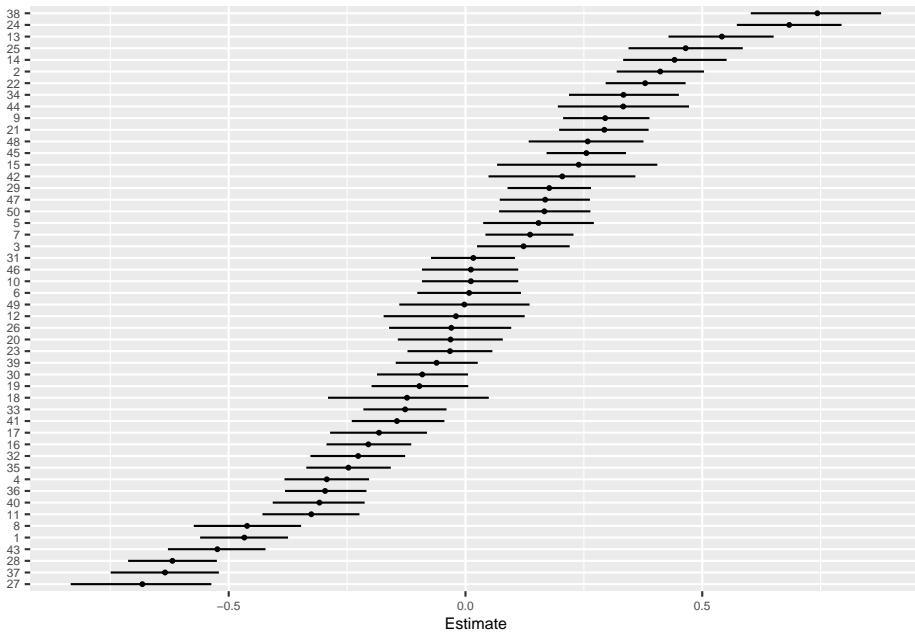
Ennustav plot - kuidas lähevad kokku mudeli ennustused reaalsete y-i andmepunktidega

```
pr <- predict(m2) %>% cbind(iris)
ggplot(pr, aes(Sepal.Length, Estimate, color = Species)) +
  geom_pointrange(aes(ymin = Q2.5, ymax = Q97.5)) +
  geom_abline(lty = 2) +
  coord_cartesian(xlim = c(4, 8), ylim = c(4, 8))
```



Igale andmepunktile – kui palju erineb selle residuaal 0-st kui hästi ennustab mudel just seda andmepunkti. Ruumi kokkuhoiiks plotime välja ainult irise valiku 50-st andmepunktist.

```
set.seed(69)
as_data_frame(residuals(m2)) %>%
  sample_n(50) %>%
  ggplot(aes(x = reorder(seq_along(Estimate), Estimate), y = Estimate)) +
  geom_pointrange(aes(ymin = Q2.5, ymax = Q97.5), fatten = 0.1) +
  coord_flip() +
  theme(text = element_text(size = 8), axis.title.y = element_blank()) +
  xlab("Residuals (95 CI)")
#> Warning: `as_data_frame()` is deprecated, use `as_tibble()` (but mind the new semantics)
#> This warning is displayed once per session.
```



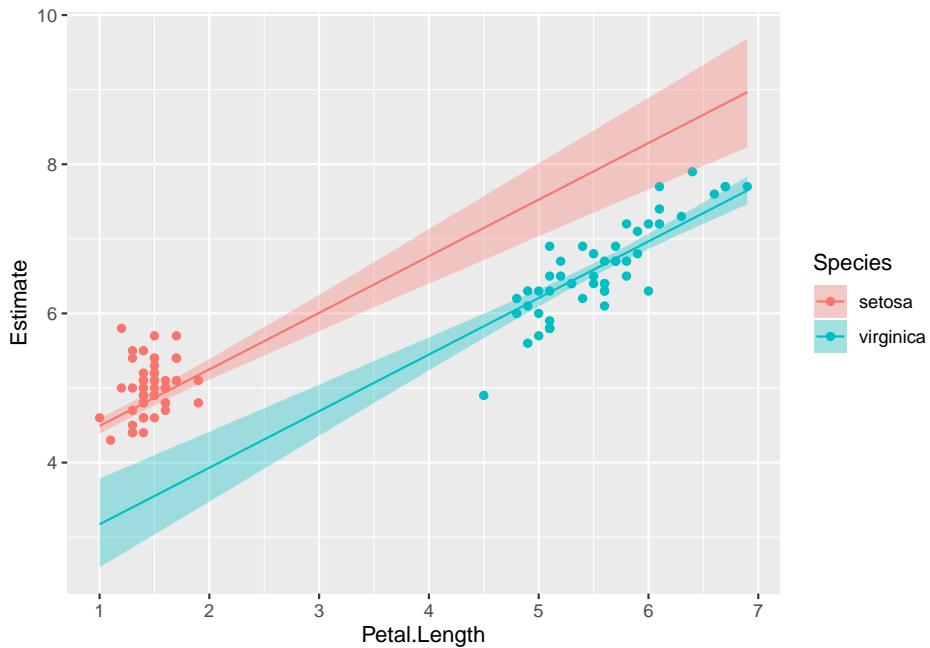
Ok, isendid nr 15 ja 44 paistavad olema vastavalt palju suurema ja väiksema Sepal Lengthiga kui mudel ennustab. Võib küsida, miks?

Nüüd plotime usaldusintervalli mudeli fitile ('keskmisele' Y väärusele igal määratud X-i vääruse sel), mitte Y-ennustusele andmepunkti kaupa. Selleks on hea fitted() funktsioon. Me ennustame m2 mudelist vastavalt newdata parameetrväärtustele. Kui me newdata argumendi tühjaks jäätame, siis võtab fitted() selleks automaatselt algse iris tabeli (ehk valimi väärused).

```
predict_interval_brms2f <- fitted(m2, newdata = newx, re_formula = NULL) %>%
  cbind(newx, .)
head(predict_interval_brms2f)

#>   Petal.Length Sepal.Width Species Estimate Est.Error Q2.5 Q97.5
#> 1      1.00     3.06 setosa    4.49  0.0542 4.38 4.59
#> 2      1.04     3.06 setosa    4.52  0.0535 4.41 4.62
#> 3      1.08     3.06 setosa    4.55  0.0529 4.45 4.65
#> 4      1.12     3.06 setosa    4.58  0.0524 4.48 4.68
#> 5      1.16     3.06 setosa    4.61  0.0520 4.51 4.71
#> 6      1.20     3.06 setosa    4.64  0.0518 4.54 4.74

ggplot(data = predict_interval_brms2f, aes(x = Petal.Length, y = Estimate, color = Species)) +
  geom_point(data = no_versicolor, aes(Petal.Length, Sepal.Length, color = Species)) +
  geom_line() +
  geom_ribbon(aes(ymax = Q2.5, ymin = Q97.5, fill = Species), alpha = 1/3, colour = NA) +
  scale_x_continuous(breaks = 0:10)
```



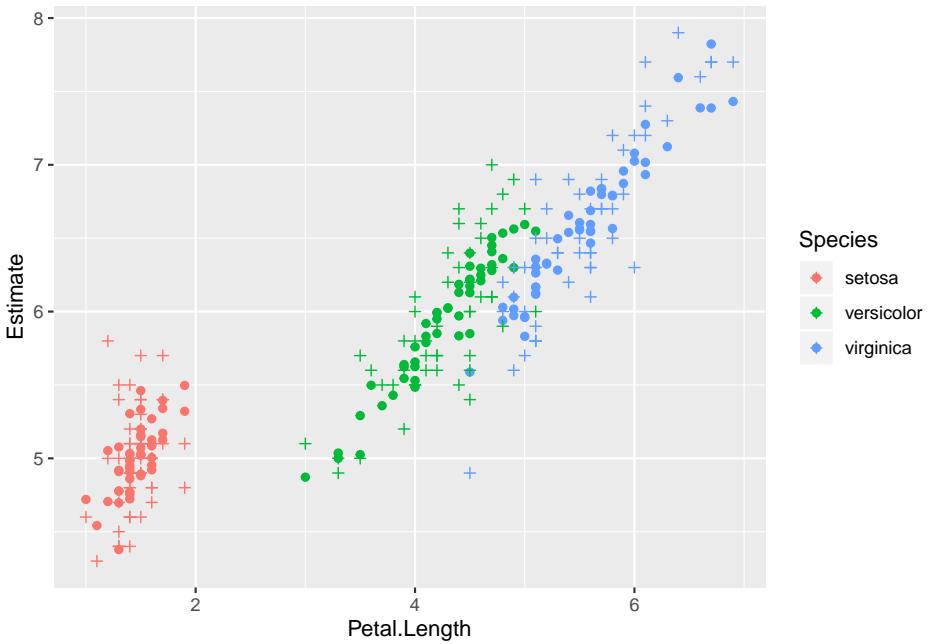
Mudeli genereeritud andmed ja valimiandmed mõõtmisobjekti (subjekti e taimeisendi) kaupa. See on sisuliselt posterior predictive plot (vt eespool).

```
predicted_subjects_brms <- predict(m2) %>% cbind(iris, .)
```

`predict()` arvutab mudeli põhjal uusi Y muutuja andmepunkte. Võib kasutada ka väljamöeldud andmete pealt Y väärustuse ennustamiseks (selleks tuleb anda ette andmeraam kõigi X-muutujate väärustega, mille pealt tahetakse ennustusi).

Punktid on ennustused ja ristikesed on valimiandmed

```
ggplot(data = predicted_subjects_brms, aes(x = Petal.Length, color = Species)) +
  geom_point(aes(y = Estimate)) +
  geom_point(aes(y = Sepal.Length), shape = 3)
```



21.1.13 Alternatiiv – ansamblieennustus

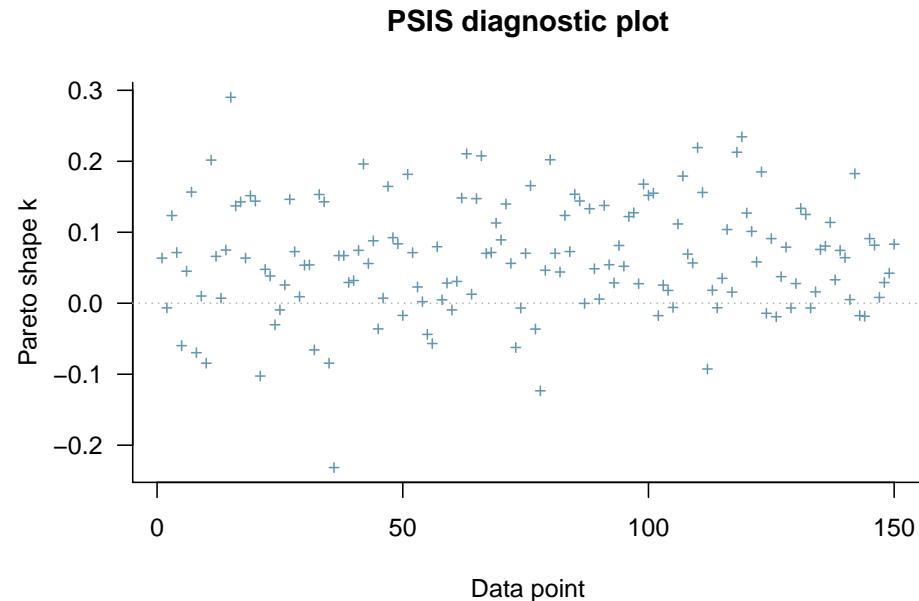
Kuna meil on 2 mudelit (m_2 ja m_3) mis on pea võrdselt eelistatud, siis genereerime ennustused mõlemast (mudelite ansamblist) proportsionaalselt nende waic skooridega. See ennustus kajastab meie mudeldamistööd tervikuna, mitte ühte "parimat" mudelit ja seega võib loota, et annab paremini edasi meie mudeldamises peituvalt ebakindlust.

```
pp_a <- pp_average(m2, m3, weights = "waic", method = "predict") %>%
  as_tibble() %>%
  bind_cols(iris)
ggplot(data = pp_a, aes(x = Petal.Length, color = Species)) +
  geom_point(aes(y = Estimate)) +
  geom_point(aes(y = Sepal.Length), shape = 3)
```

21.2 Mudeli eelduste kontroll

Pareto k otsib nn mõjukaid (influential) andmepunkte.

```
loo_m2 <- loo(m2)
plot(loo_m2)
```



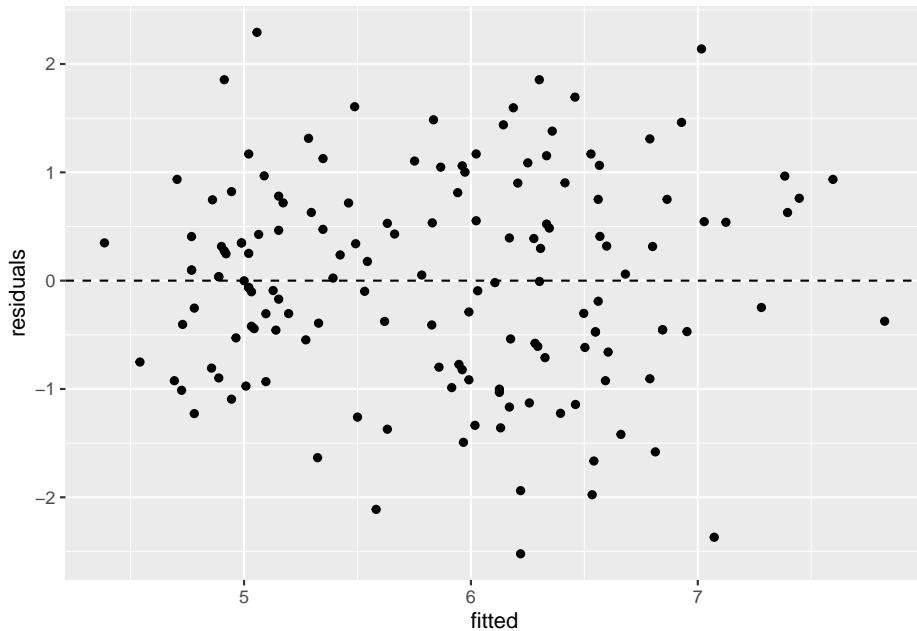
Kui paljud andmepunktid on kahtlaselt mõjukad?

```
pareto_k_table(loo_m2)
#>
#> All Pareto k estimates are good (k < 0.5).
```

21.2.1 Plotime residuaalid

`resid()` annab residuaalid vektorina. Kõigepealt plotime residuaalid fititud (keskmiste) Y väärustele vastu:

```
resid <- residuals(m2, type = "pearson")
fit <- fitted(m2)
ggplot() +
  geom_point(aes(x = fit[, "Estimate"], y = resid[, "Estimate"])) +
  geom_hline(yintercept = 0, lty = "dashed") +
  labs(x = "fitted", y = "residuals")
```

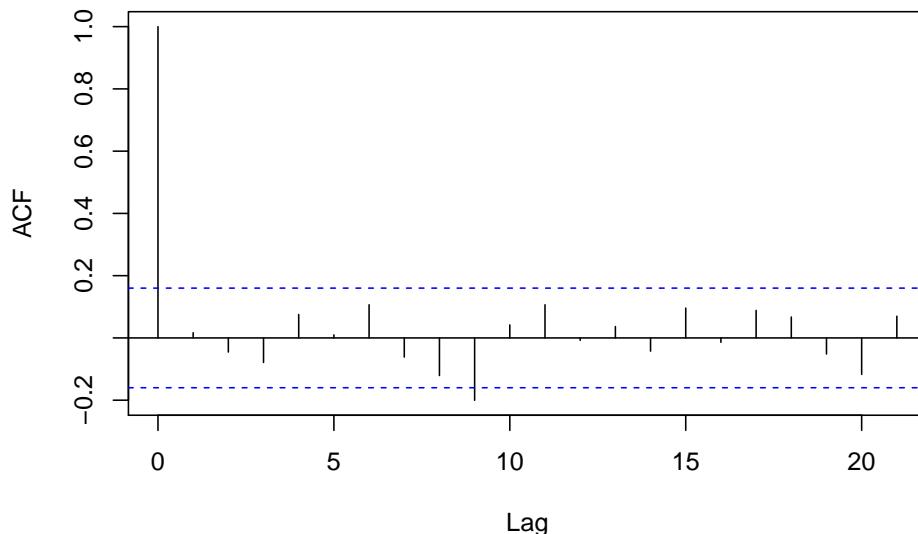


`type = "pearson"` annab standardiseeritud residuaalid $R = (Y - Y_p)/SD(Y)$, kus $SD(Y)$ on hinnang Y-muutuja SD-le. alternatiiv on `type = "ordinary"`, mis annab tavalised residuaalid.

Residuals vs fitted plot testib lineaarsuse eeldust - kui `.resid` punktid jaotuvad ühtlaselt nulli ümber, siis mudel püüab kinni kogu süstemaatilise varieeruvuse teie andmetest ja see mis üle jäab on juhuslik varieeruvus.

Vaatame diagnostilist plotti autokorrelatsioonist residuaalide vahel.

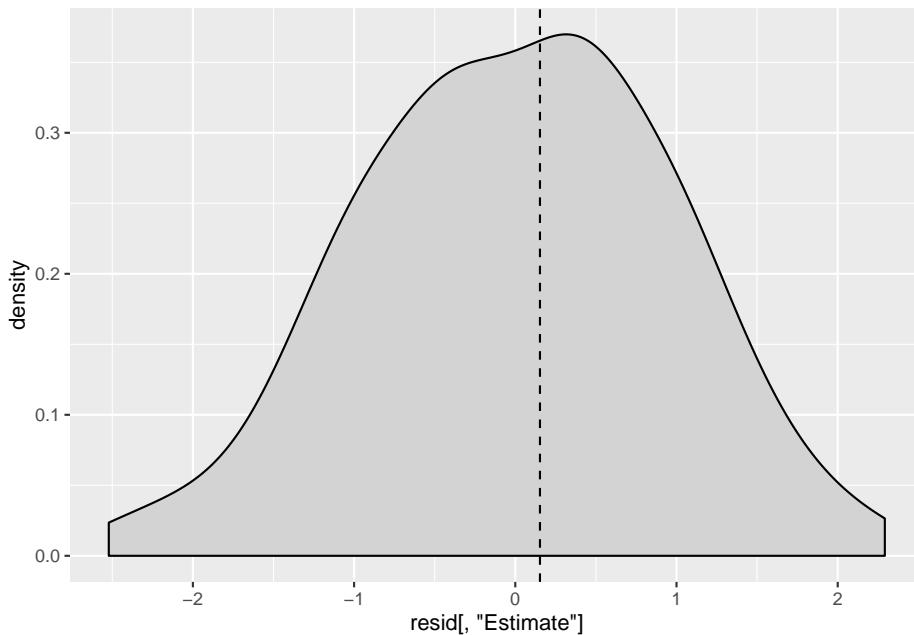
```
acf(resid[,1])
```

Series resid[, 1]

Residuaalide autokorrelatsioonid on madalad - seega kõik paistab OK ja andmepunktide sõltumatus on tagatud.

Siin on residuaalide histogramm:

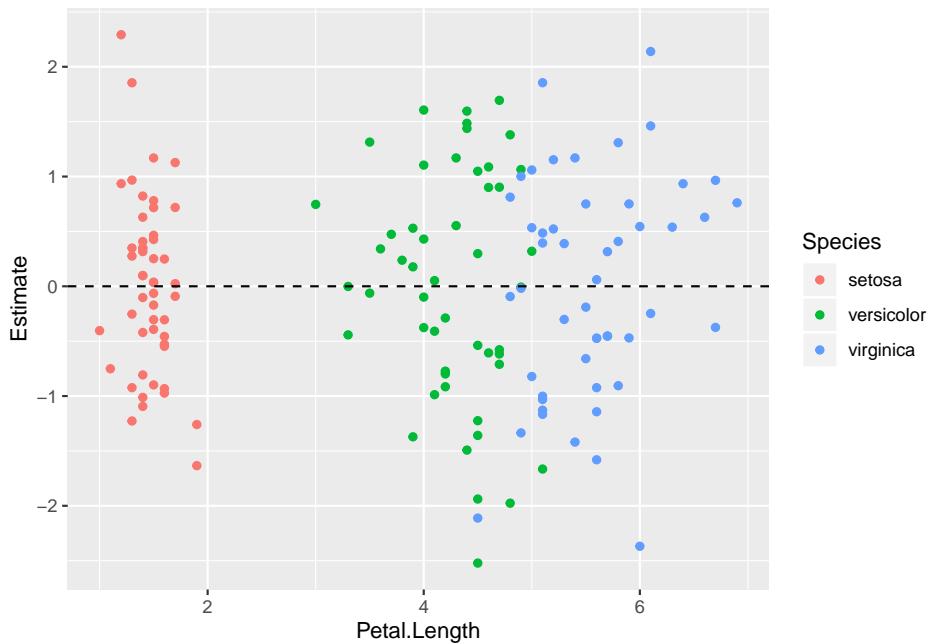
```
ggplot(data = NULL) +
  geom_density(aes(x = resid[, "Estimate"]), fill = "lightgrey") +
  geom_vline(xintercept = median(resid), linetype = "dashed")
```



Residuaalid on sümmeetrilise jaotusega ja meedian residuaal on peaaegu null. See on kõik hea.

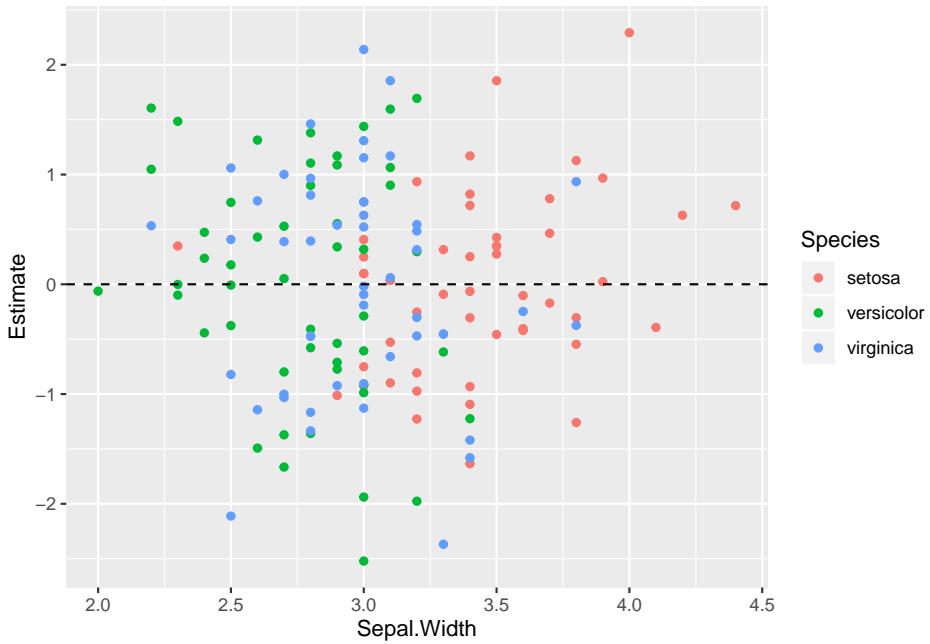
Ja lõpuks plotime standardiseeritud residuaalid kõigi x-muutujate vastu. Kõigepealt ühendame resid vektori irise tabeliga, et oleks mugavam plottida. residuaalid standardhälbe ühikutes saab ja ka tuleks plottida kõigi x-muutujate suhtes.

```
iris2 <- iris %>% cbind(resid)
ggplot(iris2, aes(Petal.Length, Estimate, color = Species)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed")
```

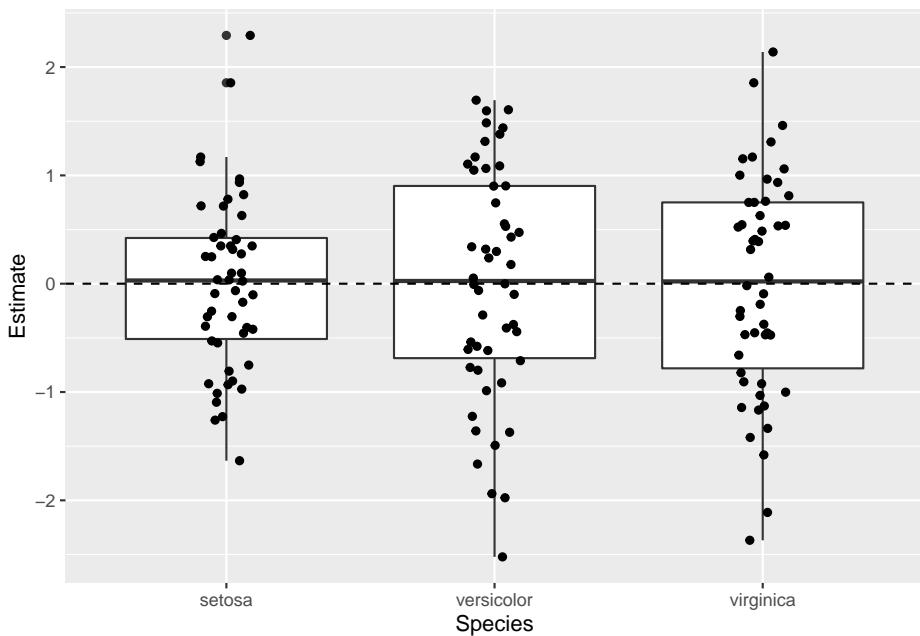


Tsiteerides klassikuid: “Pole paha!”. Mudel ennustab hästi, aga mõne punkti jaoks on ennustus 2 sd kaugusele.

```
ggplot(iris2, aes(Sepal.Width, Estimate, color = Species)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed")
```



```
ggplot(iris2, aes(Species, Estimate)) +
  geom_boxplot() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_jitter(width = 0.1)
```



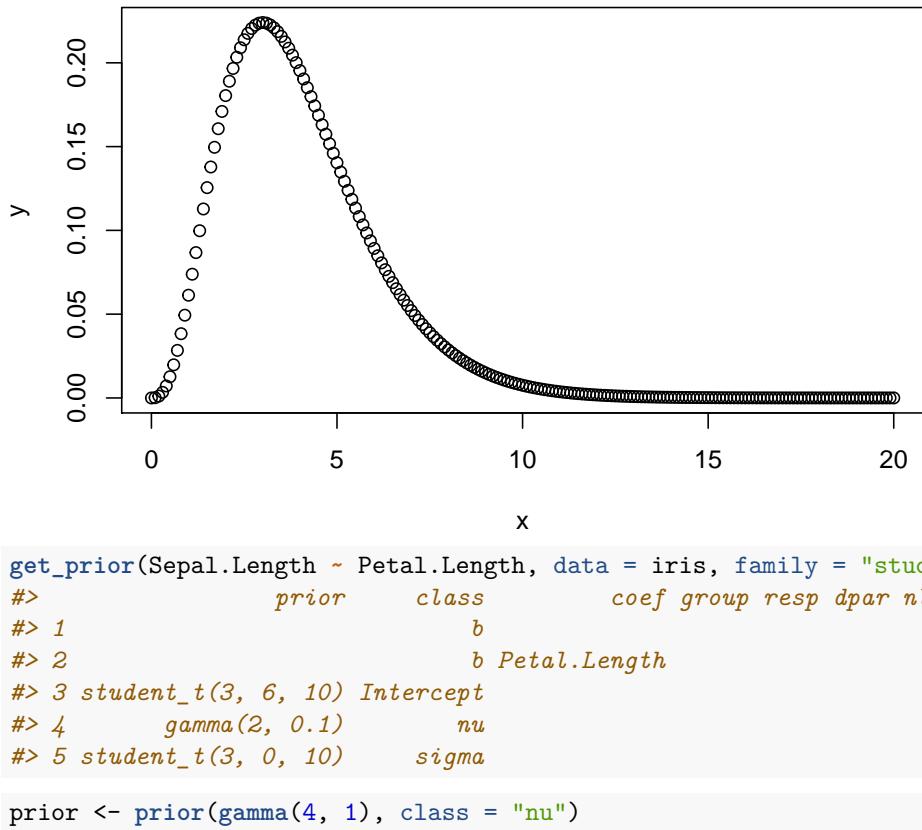
Chapter 22

Brms mudelid

22.1 Robustne lineaarne regressioon

Kasutame dnorm likelihoodi asemel studenti t jaotust. Selle jaotuse õlad on reguleeritavalts kõrgemad ja nende alla mahuvad paremini outlierid. Ōlgade kõrgust reguleerib parameeter nu (1 - Inf), mille väiksemad väärtsused (<10) annavad laiad ja kaitse outlierite vastu. Me anname nu-le gamma priori. Sellel prioril on omakorda 2 parameetrit, *shape* ja *scale*. Kui shape = 4 ja scale = 1, siis saame kitsa priori, mis eelistab nu väärtsusi, mis soosivad laiu õlgu ja robustset regressiooni.

```
x = seq(from = 0, to = 20, by = .1)
y = dgamma(x, shape = 4, scale = 1)
plot(y ~ x)
```



robust_m1 on studenti likelihooldiga, mille õlgade kõrgus fütitakse adaptiivselt andmete põhjal. robust_m2-s annane õlgade laiuse ette ja robust_m3 on mitte-robustne kontroll tavalsele normaaluse likelihooldiga.

```

robust_m1 <- brm(Sepal.Length ~ Petal.Length,
                    data = iris,
                    family = "student",
                    prior = prior)
robust_m2 <- brm(bf(Sepal.Length~Petal.Length, nu = 4),
                    data = iris,
                    family = student,
                    prior = c(prior(normal(0, 100), class = Intercept),
                            prior(normal(0, 10), class = b),
                            prior(student_t(5, 0, 5), class = sigma)))
robust_m3 <- brm(Sepal.Length~Petal.Length,
                    data = iris,
                    family = "gaussian")
write_rds(robust_m1, path = "data/robust_m1.fit")
write_rds(robust_m2, path = "data/robust_m2.fit")

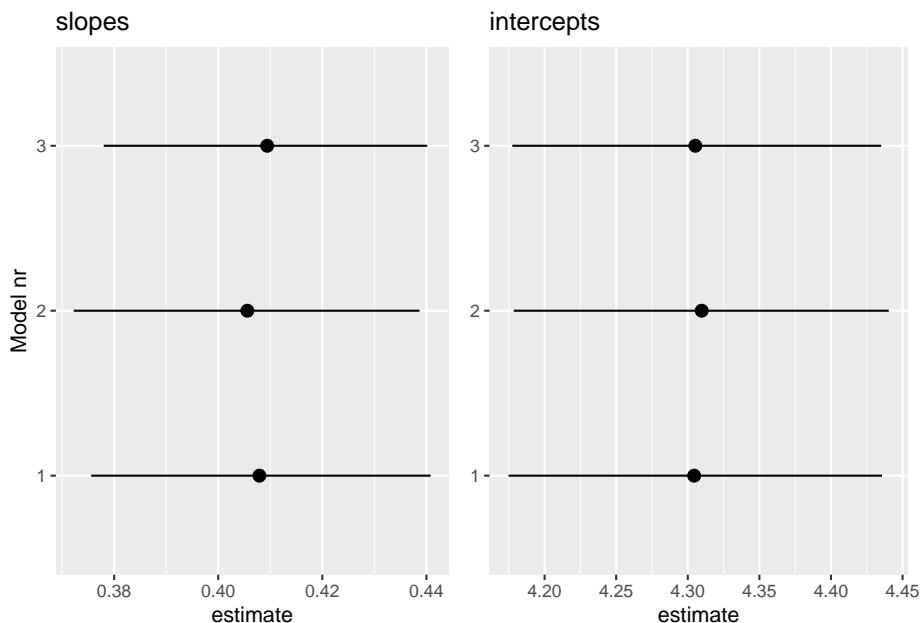
```

```

write_rds(robust_m3, path = "data/robust_m3.fit")

b_estimates <- bind_rows(tidy(robust_m1),
                          tidy(robust_m2),
                          tidy(robust_m3),
                          .id = "model_nr")
b1 <- filter(b_estimates, str_detect(term, "b_P")) %>%
  ggplot(aes(model_nr, estimate)) +
  geom_pointrange(aes(ymin = lower, ymax = upper)) +
  coord_flip() +
  labs(x = "Model nr", title = "slopes")
b2 <- filter(b_estimates, str_detect(term, "b_I")) %>%
  ggplot(aes(model_nr, estimate)) +
  geom_pointrange(aes(ymin = lower, ymax = upper)) +
  coord_flip() +
  labs(x = NULL, title = "intercepts")
gridExtra::grid.arrange(b1, b2, nrow = 1)

```



Kolme mudeli lõikepunktid ja tõusunurgad on sisuliselt võrdsed ja sama täpsusega hinnatud. Seega ei tee robustne mudel vähemal halba, kui meil on enam-vähem normaalsed andmed.

Proovime ka robuststet versiooni 2 grupi võrdlusest (vastab t testile, kus kahe grupi sd-d hinnatakse eraldi)

```

no_versicolor <- filter(iris, Species != "versicolor")
get_prior(bf(Sepal.Length ~ Species, sigma ~ Species),
           data = no_versicolor,
           family = "student")
#>          prior      class      coef group resp  dpar npar
#> 1            b
#> 2            b Speciesvirginica
#> 3 student_t(3, 6, 10) Intercept
#> 4      gamma(2, 0.1)    nu
#> 5            b
#> 6            b Speciesvirginica      sigma
#> 7 student_t(3, 0, 10) Intercept      sigma
#>   bound
#> 1
#> 2
#> 3
#> 4
#> 5
#> 6
#> 7

prior <- c(prior(gamma(4, 1), class = "nu"),
           prior(normal(0, 4), class = "b"))

robust_t_test1 <- brm(bf(Sepal.Length ~ Species, sigma ~ Species),
                       data = no_versicolor,
                       prior = prior,
                       family = "student")
write_rds(robust_t_test1, path = "data/robust_t_test1.fit")

tidy(robust_t_test1)
#>          term estimate std.error lower upper
#> 1      b_Intercept  5.002   0.0505  4.916  5.086
#> 2      b_sigma_Intercept -1.175   0.1281 -1.383 -0.970
#> 3      b_Speciesvirginica  1.557   0.1032  1.387  1.726
#> 4      b_sigma_Speciesvirginica  0.577   0.1694  0.298  0.846
#> 5            nu  6.113   2.0334  3.337  9.874
#> 6            lp__ -79.991  1.5766 -82.954 -78.063

```

b_Intercept on hinnang 1. grupi keskväärtusele (algseks skaalas)

b_Speciesvirginica on hinnag efekti suurusele, ehk 2. grupi erinevusest esimesest grupist (algseks skaalas)

b_Intercept + b_Speciesvirginica annab 2. grupi keskväärtuse.

b_sigma_Intercept on naturaallogaritm 1. grupi sd-st.

```
exp(-1.175)
#> [1] 0.309
```

Tegelik sigma on 0.3

b_sigma_Speciesvirginica on logaritm 2. grupi (*I. virginica*) sd erinevusest esimesest grupist (ehk efekti suurus).

```
exp(-1.175 + 0.577)
#> [1] 0.55
```

Seega saab algses skaalas sd-d nii:

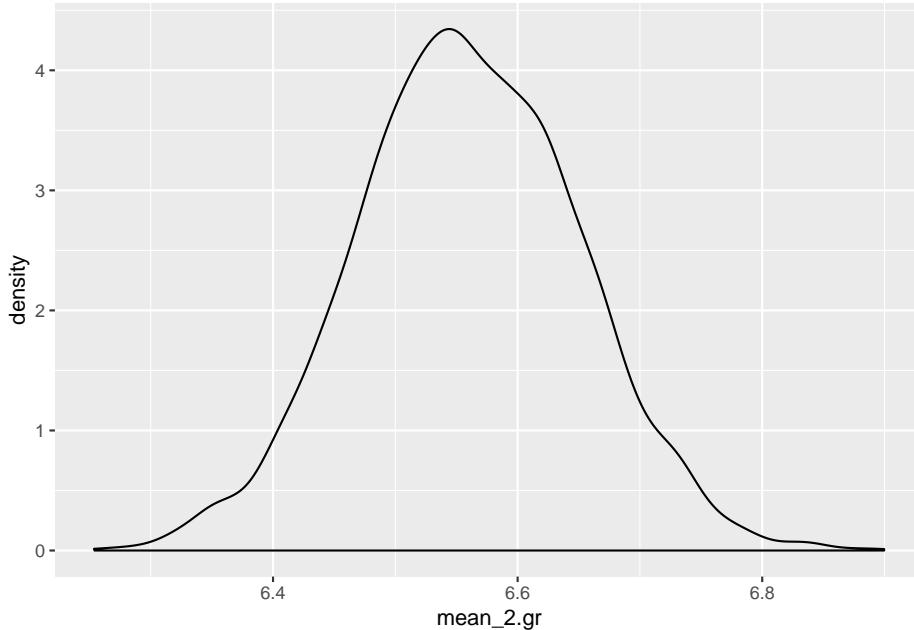
$$\exp(b_{\text{sigma}}_{\text{Intercept}}) = 1. \text{ grupi } \text{sd}$$

$$\exp(b_{\text{sigma}}_{\text{Intercept}}) + \exp(b_{\text{sigma}}_{\text{Speciesvirginica}}) = 2. \text{ grupi } \text{sd}$$

$$\exp(b_{\text{sigma}}_{\text{Speciesvirginica}}) = \text{sd-de erinevus}$$

Nii arvutame 2. grupi keskväärtuse posteeriori

```
r_1_df <- posterior_samples(robust_t_test1)
mean_2.gr <- r_1_df$b_Intercept + r_1_df$b_Speciesvirginica
ggplot(data = NULL) + geom_density(aes(mean_2.gr))
```



Nii saab tekitada usaldusinettevalle, mis katavad 90% jaotuse alusest kõrgeimast tihedusest (mis ei ole päris sama, mis kvantiilide meetod)

```

rethinking::HPDI(mean_2.gr, prob = 0.9)
#> [0.9 0.9]
#> 6.4 6.7

quantile(mean_2.gr, probs = c(0.05, 0.95))
#> 5% 95%
#> 6.41 6.71

```

Nii saame teada, milline osa (fraktsioon) posteeriorist on väiksem kui 6.4

```

mean(mean_2.gr < 6.4)
#> [1] 0.0372

```

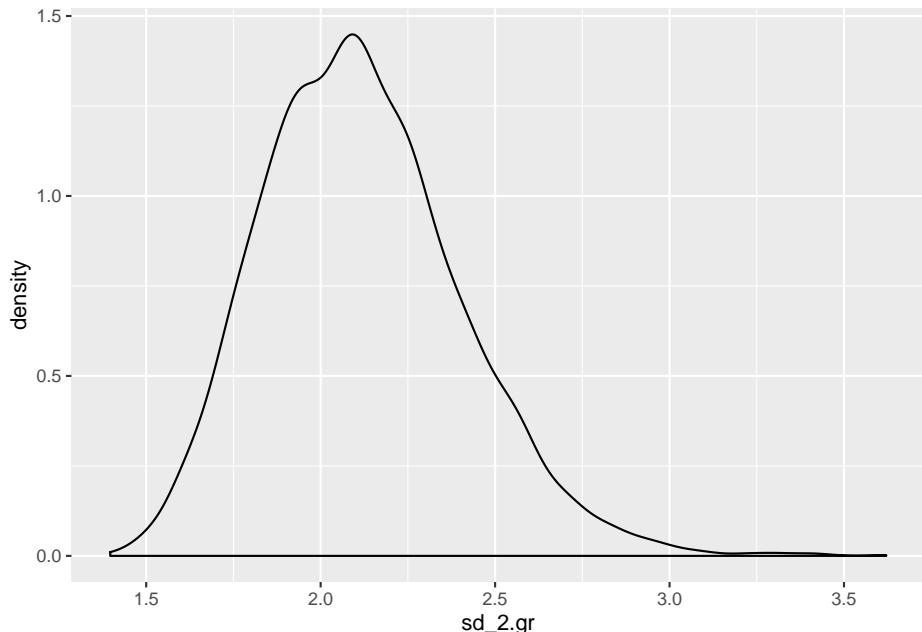
Asendades eelnevas koodis 6.4 nulliga saame bayesi versiooni ühepoolsest p väärustusest hüpoteesile, et teise gruvi keskväärtus on null.

Avaldame posteeriori 2. gruvi sd-e

```

sd_2.gr <- exp(r_1_df$b_sigma_Intercept) + exp(r_1_df$b_sigma_Speciesvirginica)
ggplot(data = NULL) +
  geom_density(aes(sd_2.gr))

```



On tavalline, et sd-de posteeriorid ei ole normaaljaotusega (selle kohta vaata lähemalt Statistical Rethinking raamatust).

```

t.test(Sepal.Length ~ Species, data = no_versicolor)
#>
#> Welch Two Sample t-test

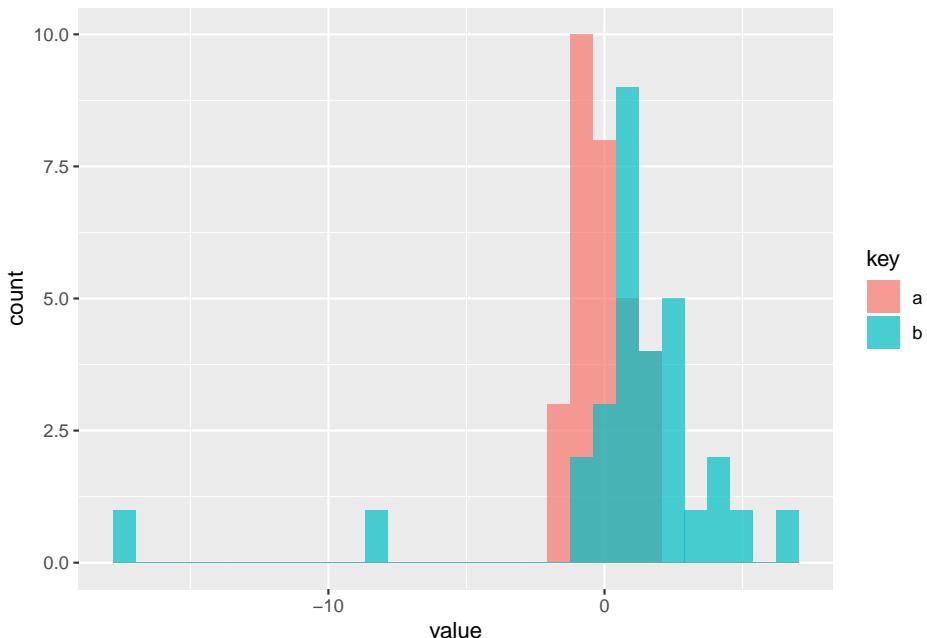
```

```
#>
#> data: Sepal.Length by Species
#> t = -15, df = 77, p-value <2e-16
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#> -1.79 -1.38
#> sample estimates:
#> mean in group setosa mean in group virginica
#> 5.01 6.59
```

Klassikalise t testi efekti suuruse CI on $-1.79 \dots -1.38$ robustse t testi oma on $-1.39 \dots -1.73$

Simuleerime siis ühe tõsiste outlieritega andmestiku, et vaadata kas meil õnnes-tub päästa efekt statistilise mitteolulisuse õnnetust saatusest. Meil on a grupis 30 andmepunkti normaaljaotusest $\mu = 0$, $sd = 1$ ja b grupis 25 andmepunkti normaaljaotusest $\mu = 1$, $sd = 1.5$, pluss 5 andmepunkti, mis mängivad outliereid.

```
set.seed(123)
df1 <- tibble(a = rnorm(30), b = c(rnorm(25, 1, 1.5), 4.3, 5.3, 7, -8.1, -17)) %>% gather()
ggplot(df1, aes(value, fill = key)) +
  geom_histogram(alpha = 0.7, position = "identity", bins = 30)
```



```
robust_t_test2 <- brm(bf(value ~ key, sigma ~ key),
                      data = df1,
```

```

family = "student",
prior = prior(gamma(4, 1), class= "nu"))
write_rds(robust_t_test2, path = "data/robust_t_test2.fit")

tidy(robust_t_test2)
#>             term  estimate std.error    lower    upper
#> 1      b_Intercept -0.0961   0.188 -0.410  0.2049
#> 2 b_sigma_Intercept -0.2372   0.192 -0.557  0.0689
#> 3      b_keyb     1.4340   0.405  0.776  2.0979
#> 4 b_sigma_keyb     0.6312   0.287  0.164  1.0979
#> 5      nu        2.8096   0.952  1.585  4.6058
#> 6      lp__    -124.5835   1.718 -128.006 -122.5280

t.test(value ~ key, data = df1)
#>
#> Welch Two Sample t-test
#>
#> data: value by key
#> t = -1, df = 32, p-value = 0.3
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#> -2.417 0.777
#> sample estimates:
#> mean in group a mean in group b
#> -0.0471      0.7729

```

Nüüd kus meil on outlieritega andmed, annab klassikaline t test efekti suurusele CI -2.41 ... 0.78 ($p = 0.3$), aga robustne t test leiab efekti üles - CI 0.78 ... 2.10 [tegelik ES oleks 1, outliereid arvestamata].

Ilma outlieriteta versioon annab $p = 0.00006$

```

set.seed(123)
a = rnorm(30)
b = c(rnorm(25, 1, 1.5))
t.test(a, b)$p.value
#> [1] 6.37e-05

```

Kui tavalline t test annab välja kahe gruvi keskmised, usaldusintervalli nende erinevusele (ehk ES-le) ja p väärtsuse, siis bayesi variant annab välja 2 gruvi keskväärtused, 2 gruvi varieeruvused andmepunktide tasemel ning kõik efekti suurused ja hüpooteesitestid, millest te suudate unistada. Selle külluse põhjus on, et hinnang iga parameeteri väärtsusele tuleb meile posteeriori ehk töenäosusjaotuse kujul. Kuna iga posteerior on meil arvutis olemas kui arvuline vektor, ja teatavasti saab vektoritega teha aritmeetilisi tehteid, siis saab ka posteerioreid omavahel liita, lahutada, astendada jms. Teoreetiliselt sisaldab posteerior kogu infot, mis meil vastava parameetri väärtsuse kohta on. Me ei vaja midagi enamat,

et teha kõiki järeldusi, mida me selle parameetri väärтuse kohta üldse teha saame. Seetõttu on bayesi versioon mitte ainult palju paindlikum kui tavaline t test, vaid selle output on ka hästi palju informatiivsem.

Igaks juhuks tuletame meealde, et tavaline t test (küll versioonis, kus võrreldavate gruppide varieeruvused on eeldatud olema identsed) on ekvivalentne lineaarse regressiooniga, mille siin fitime vähimruutude meetodiga. (Väheinformatiivsete prioritega bayesi versioon normaaljaotuse likelihoodiga annaks sellega väga sarnase fiti.)

```
lm1 <- lm(value ~ key, data = df1)
tidy(lm1)
#> # A tibble: 2 x 5
#>   term      estimate std.error statistic p.value
#>   <chr>     <dbl>     <dbl>     <dbl>    <dbl>
#> 1 (Intercept) -0.0471    0.554    -0.0850  0.933
#> 2 keyb        0.820     0.784     1.05     0.300
```

$p = 0.3$ ongi vastava t testi põhiväljund.

22.2 lognormaalne tõepärafunktsioon

Sama näide on pikemalt 14. peatükis, seal küll lahendatud rethinkingu abil.

```
library(gapminder)
library(rethinking)
g2007 <- gapminder %>%
  filter(year == 2007) %>%
  mutate(l_GDP = log10(gdpPerCap))

get_prior(gdpPerCap~lifeExp, family = "lognormal", data=g2007)
#>           prior      class     coef group resp dpar nlnpar bound
#> 1                   b
#> 2                   b lifeExp
#> 3 student_t(3, 9, 10) Intercept
#> 4 student_t(3, 0, 10) sigma

prior <- c(prior(normal(0, 10), class = "Intercept"),
          prior(normal(0, 10), class = "b"),
          prior(student(6, 0, 5), class = "sigma"))

ln_m1 <- brm(gdpPerCap ~ lifeExp, family = "lognormal", prior = prior, data = g2007, cores = 4)
write_rds(ln_m1, "ln_m1.rds")

tidy(ln_m1)
#>   term      estimate std.error      lower      upper
#> 1 b_Intercept  2.53e+00   0.3891  1.88e+00   3.162
```

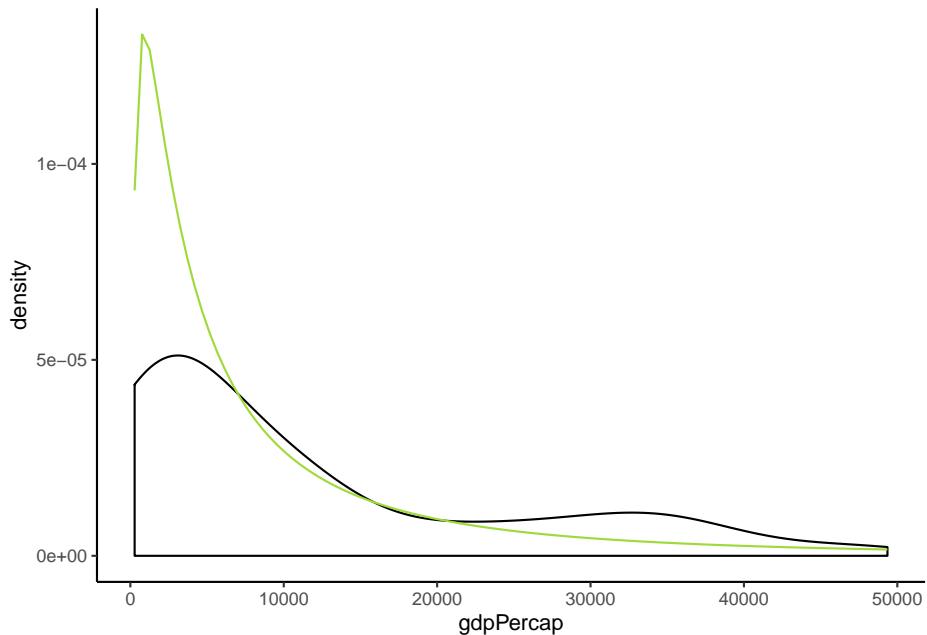
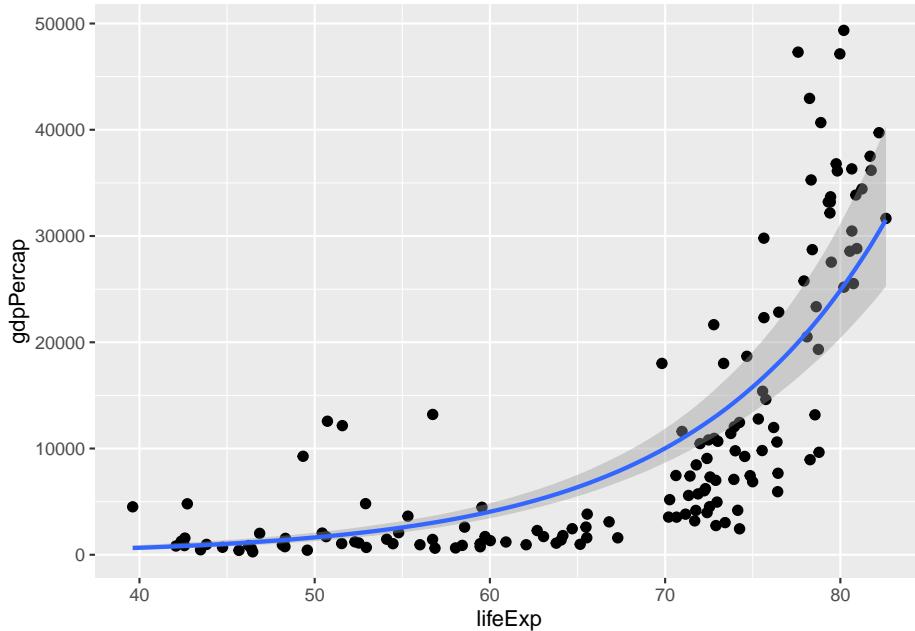
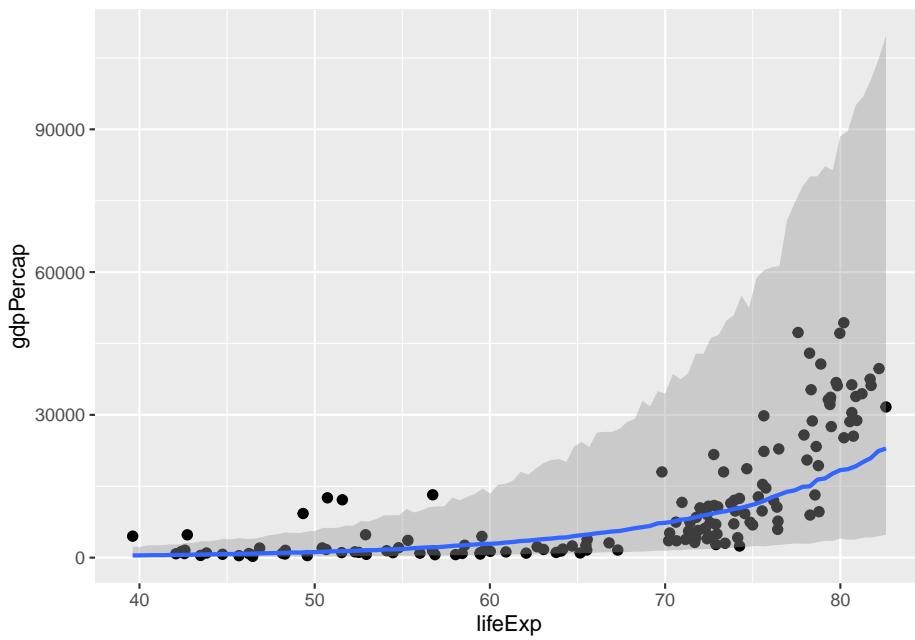


Figure 22.1: SKP-de jaotus

```
#> 2    b_lifeExp  9.09e-02   0.0057  8.15e-02   0.100
#> 3      sigma  8.08e-01   0.0504  7.30e-01   0.895
#> 4      lp__ -1.40e+03  1.2649 -1.40e+03 -1398.543
plot(marginal_effects(ln_m1), points = TRUE)
```

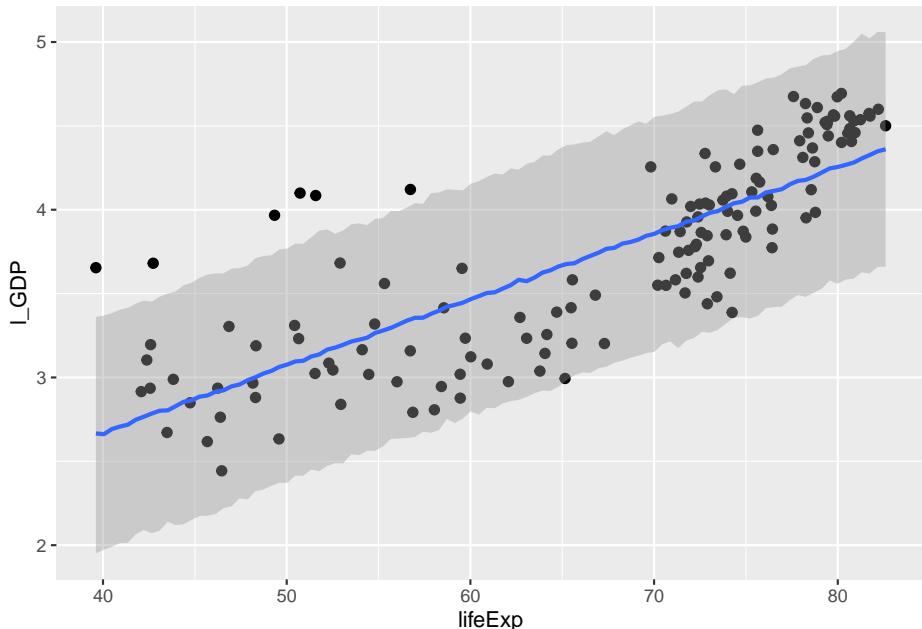


```
plot(marginal_effects(ln_m1, method = "predict"), points = TRUE)
```



```
ln_m2 <- brm(l_GDP ~ lifeExp, data=g2007, cores=4)
write_rds(ln_m2, "ln_m2.rds")
```

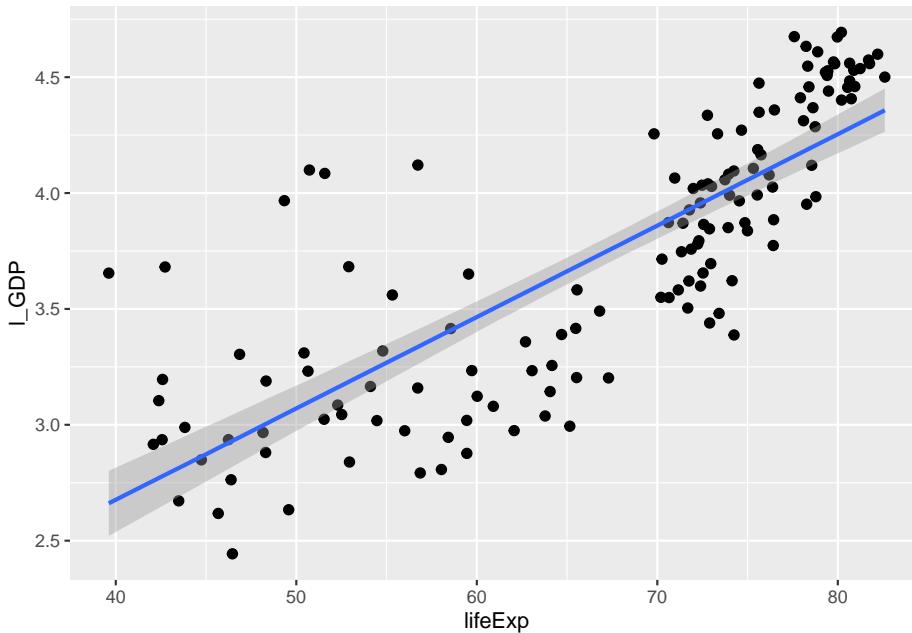
```
ln_m2 <- read_rds("data/ln_m2.rds")
plot(marginal_effects(ln_m2, method="predict"), points=TRUE)
```



```
bayes_R2(ln_m1)
#>   Estimate Est.Error Q2.5 Q97.5
#> R2     0.587    0.0767 0.429 0.727

bayes_R2(ln_m2)
#>   Estimate Est.Error Q2.5 Q97.5
#> R2     0.652    0.0277 0.59  0.698

plot(marginal_effects(ln_m2), points=TRUE)
```



22.3 Puuduvate andmete imputatsioon

Regressioonimudelite fittimisel kasutatakse ainult vaatlusi, kus esinevad väärustused kõigis mudelisse pandud muutujates. Seega, kui meil on palju muutujaid, milles igaühes puuduvad juhuslikult mõned väärustused, siis kaotame kokkuvõttes enamuse oma valimist. Aitab puuduvate andmete imputatsioon, mis tegelikult tähendab, et me fitime iga puuduvaid andmeid sisaldaava muutuja eraldi regressioonimudelis kõigi teiste muutujate vastu.

Eriti vajalik, kui andmed ei puudu juhuslikult!

Viskame irise andmestiku kahest tulbast välja 1/4 andmepunkte, aga mitte juhuslikult vaid kõik madalamad väärustused. Selline suunatud tegevus kallutab (ehk suunab kindlas suunas) oluliselt mudeldamise tulemusi

```
quantile(iris$Petal.Length)
#>   0%  25%  50%  75% 100%
#> 1.00 1.60 4.35 5.10 6.90
iris_na <- iris
iris_na$Sepal.Length[iris_na$Sepal.Length < 5] <- NA
iris_na$Petal.Length[iris_na$Petal.Length < 1.6] <- NA

lm(Petal.Length ~ Sepal.Length, data = iris) %>% tidy()
#> # A tibble: 2 x 5
#>   term        estimate std.error statistic p.value
#>   <fct>     <dbl>    <dbl>     <dbl>    <dbl>
```

```
#>   <chr>      <dbl>    <dbl>    <dbl>    <dbl>
#> 1 (Intercept) -7.10    0.507   -14.0   6.13e-29
#> 2 Sepal.Length  1.86    0.0859   21.6   1.04e-47

lm(Petal.Length ~ Sepal.Length, data = iris_na) %>% tidy()
#> # A tibble: 2 x 5
#>   term      estimate std.error statistic p.value
#>   <chr>      <dbl>    <dbl>    <dbl>    <dbl>
#> 1 (Intercept) -4.19    0.609   -6.89   4.33e-10
#> 2 Sepal.Length  1.43    0.0976   14.6   4.53e-27
```

imputeerime enne mudeli fittimist kasutades multiple imputation meetodit mice paketist <https://stefvanbuuren.name/mice/>. Siin imputeerime iga puuduva väärtsuse kasutades kõigi teiste parameetrite väärtsusi, ja me teeme seda 5 korda.

```
library(mice)
imp <- mice(iris_na, m = 5, print = FALSE)
```

Meil on nüüd 5 imputeeritud andmesetti. Me saadame need kõik brms-i.

Siin kasutame mice() tema vaikeväärustel, kuid mice pakett on tegelikult vägagi rikkalik imputatsioonimasin, mille helpi ja tutoorialeid tuleks kindlasti enne lugeda, kui oma andmeid imputeerima asuda. Lisaks, see raamat on tervenisti pühendatud imputatsioonile: <https://stefvanbuuren.name/fimd/>

```
iris_imp1 <- brm_multiple(Petal.Length ~ Sepal.Length, data = imp)
write_rds(iris_imp1, path = "data/iris_imp1.fit")
```

Saame tavaliise fitiobjekti, kus on 5 alammudeli posterioorid. Kõik juba koos.

```
tidy(iris_imp1)[,1:3]
#>           term estimate std.error
#> 1 b_Intercept   -7.692    0.5543
#> 2 b_Sepal.Length  1.948    0.0929
#> 3 sigma          0.843    0.0523
#> 4 lp__          -192.574   3.2501
```

Tõepoolest, süstemaatiliselt rikutud andmetest on imutatsiooni abil võimalik täitsa head ennustust tagasi saada!!!

Imputatsioon otse brms-is

See töötab küll irise peal halvemini kui mice!

```
bform <- bf(Petal.Length | mi() ~ mi(Sepal.Length)) +
bf(Sepal.Length | mi() ~ Sepal.Width + Petal.Width + Species + mi(Petal.Length)) + s()
```

```
iris_imp2 <- brm(bform, data = iris_na)
write_rds(iris_imp2, path = "data/iris_imp2.fit")
```

Kasutades mi() funktsiooni peame siin eksplitsiitselt ütlema, milliseid muutujaid soovime imputeerida. Me imputeerime Petal.Length-i, kasutades NA-de väärustute prediktorina Sepal.Length-i, mis on omakorda imputeeritud. Sepal.Length-i imputeerime omakorda Sepal.Width, Petal.width jne järgi.

```
bform <- bf(Petal.Length | mi() ~ mi(Sepal.Length)) +
  bf(Sepal.Length | mi() ~ Species) + set_rescor(FALSE)
iris_imp3 <- brm(bform, data = iris_na)
write_rds(iris_imp3, path = "data/iris_imp3.fit")

tidy(iris_imp2) %>% head()
#>           term estimate std.error lower upper
#> 1 b_PetalLength_Intercept -5.5690  0.4917 -6.204 -4.8129
#> 2 b_SepalLength_Intercept  2.6008  0.3255  2.115  3.1246
#> 3 b_SepalLength_Sepal.Width  0.3074  0.0855  0.164  0.4297
#> 4 b_SepalLength_Petal.Width  0.0679  0.1515 -0.130  0.3330
#> 5 b_SepalLength_Speciesversicolor -0.2201  0.1595 -0.482  0.0172
#> 6 b_SepalLength_Speciesvirginica -0.4502  0.2319 -0.856 -0.0937

tidy(iris_imp3) %>% head()
#>           term estimate std.error lower upper
#> 1 b_PetalLength_Intercept -4.601   0.5864 -5.56 -3.650
#> 2 b_SepalLength_Intercept  5.168   0.0915  5.01  5.316
#> 3 b_SepalLength_Speciesversicolor  0.781   0.1157  0.59  0.971
#> 4 b_SepalLength_Speciesvirginica  1.449   0.1155  1.26  1.645
#> 5 bsp_PetalLength_miSepal.Length  1.488   0.0945  1.33  1.642
#> 6 sigma_PetalLength        0.705   0.0494  0.63  0.791
```

22.4 Binoomjaotusega mudelid

$$y \sim \text{Binomial}(n, p)$$

Me teeme n katset ja kodeerime iga eduka katse 1-ga ja mitteeduka katse 0-ga. Kui $n=1$, siis y on ühtedest ja nullidest koosnev vektor (muutuja) ja p on tõenäosus, et suvaline katse annab tulemuseks 1-e. Eeldades logistilist transformatsiooni on siin tegu logistilise regressiooniga. Kui $n > 1$ (ja ikka eeldades logistilist transformatsiooni), siis on tegu aggregeeritud binoomse logistilise regressiooniga. Me lahendame allpool need mõlemad.

22.4.1 Logistiline regressioon

Tavalises lineaarses regressioonis on tavapärane, et kuigi me ennustame pidevat y -muutujat, on kas osad või kõik X -muutujad mittepidevad. Sellest pole kurja, meie mudelid jooksevad nii pidevate kui mittepidevate (binaarsete) prediktoritega. (Binaarsed muutujad võivad omada kaht diskreetset väärust, mida kodeerime 1 ja 0-na) Me eeldame küll, et y -muutuja on normaalne, aga ei eelda midagi sellist x -muutujate ehk prediktorite kohta. Samuti on lubatud prediktorite mitte-lineaarsed funktsionid, nagu $X_1 X_2$ või X^2 , senkaua kui regressioonivõrrandi parameetrid (a, b_1, \dots, b_n) on lineaarsetes additiivsetes suhetes.

Aga kuidas käituda, kui meie poolt ennustataval Y -muutujal on vaid kaks võimalikku väärust, 0 ja 1, ning ta on binoomjatusega? Kui me püüame ennustada binaarse y -muutuja oodatavaid väärusi tõenäosustena, ehk 1-de arvu suhet katsete koguarvu, siis tavaline lineaarne regressioon ei garanteeri, et ennustused jäädvad 0 ja 1 vahele, ehk tõenäosuste skaalasse.

Kui eespool õppisime transformeerima andmeid, et paremini täita regressiooni eeldusi (lineaarsust ja normaalsust), siis selleks, et suruda mudeli ennustused tõenäosusskaalasse ei transformeeriri me mitte andmeid, vaid mudeli võrrandit ennast. Selliste transformeeritud mudelite ehk GLM-ide (*Generalized Linear Model*) levinuim näide on logistilise regressiooni mudel. Logistilises regressioonis ei modelleeri me mitte otse y väärustusi (1 ja 0) erinevatel x -i väärustel, vaid tõenäosust $P(Y = 1 | X)$ [loe: tõenäosus, et $Y = 1$, eeldades kindlat x -i väärust].

Logistiline regressioon kasutab logistiklist transformatsiooni, mis näiteks funktsionile $y = a + bx$ on

$$P(Y = 1 | X) = \frac{\exp(a + bx)}{1 + \exp(a + bx)}$$

$\exp(a)$ tähendab "e astmes a", kus e on Euleri arv, ehk arv, mille naturaal-logaritm = 1 (seega on e naturaal-logaritmi alus). e on umbes 2.71828 ja selle saab valemist $(1 + 1/n)^n$, kui n läheneb lõpmatusele.

Logistiline transformatsioon viib lineaarse regressiooni tavapärasest y -muutuja skaalast $[-\infty, +\infty]$ tõenäosuste skaalasse $[0, 1]$, andes sirge asemel S-kujulise kurvi, mis läheneb asümptootiliselt ühelt poolt 0-le ja teiselt poolt 1-le. Logistilise funktsiooni põõrdväärthus on logit funktsioon, mis annab "odds-i" ehk shansi ehk kihlveosuhte tõenäosusele p : $odds = \frac{p}{1-p}$. Tõenäosuse p logit ehk $\text{logit}(p)$ on sama, mis $\log(\text{odds})$:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p)$$

$y = a + bx$ mudeli korral tavalises meetrilises skaalas on $odds$ exponent fititud lineaarsest mudelist:

$$odds = \frac{P(Y = 1 | X)}{1 - P(Y = 1 | X)} = \exp(a + bx)$$

ja ekvivalentselt

$$\log(odds) = \text{logit}(p) = a + bx$$

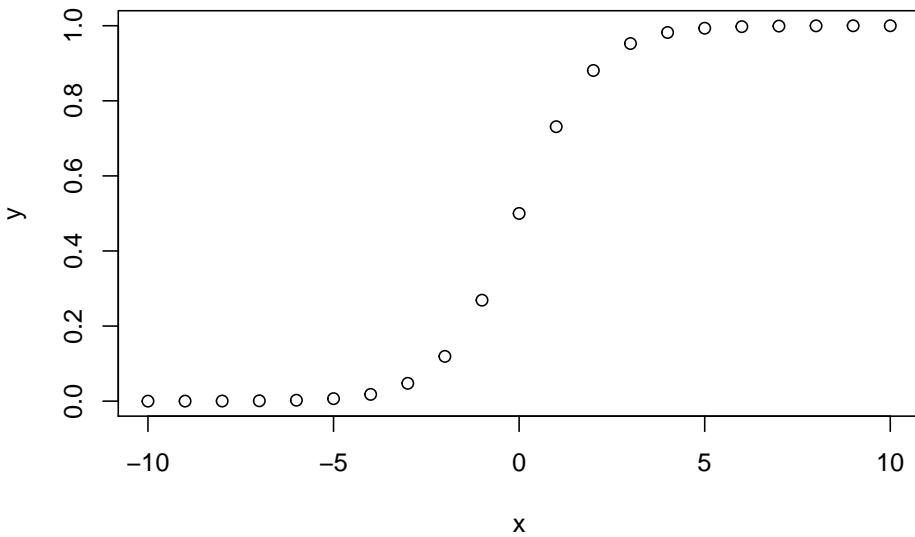
Matemaatiliselt pole vahet, kas me transformeerime prediktorid logistilise funktsiooniga või ennustatava muutuja logit funktsiooniga – need on sama asja erinevad kirjeldused.

Kuidas suhtuvad *odds*-d tõenäosustesse? Näiteks tõenäosus 0.2 (20%) tähendab, et $odds = 0.2/(1 - 0.2) = 1/4$ ehk üks neljale ja tõenäosus 0.9 tähendab, et $odds = 0.9/(1 - 0.9) = 9$ ehk üheksa ühele. *Odds*-e kasutavad näiteks hipodroomid, sest nii on mänguril lihtne näha, et kui kihlveokontori poolt mingile hobusele omistatud odds on näiteks üks nelja vastu ja ta maksab kihlveo sõlmimisel 1 euro, siis ta saab võidu korral 4 eurot kasu (ehk 5 eurose kupüüri). Logaritm *odds*-idest ongi logit transformatsioon, mille pöördväärthus on omakorda logistiline transformatsioon!

Suvalise arvu α logistiline funktsioon on logiti pöördväärthus:

$$\text{logit}^{-1}\alpha = \text{logistic}(\alpha) = \frac{\exp(\alpha)}{1 + \exp(\alpha)}$$

```
x <- -10:10
y <- exp(x) / (1+exp(x))
plot(y~x)
```



Kui me logistilise regressiooniga fititud mudeli $y = a + bx$ korral muudame x-i väärust ühe ühiku võrra, siis muutub *log-odds* b võrra, mis on sama, mis õelda, et *odds* muutub $\exp(b)$ võrra. Samas b ei vasta $P(Y = 1 | X)$ muutusele X-i muutumisel ühe ühiku võrra. See, kui kiiresti $P(Y = 1 | X)$ muutub, sõltub X-i väärustest. Siiski, semikaua kuni $b > 0$, kaasneb X-i kasvuga alati tõenäosuse $P(Y = 1)$ kasv (ja vastupidi).

Kahe tõenäosuse logitite vahe on sama, mis logaritm *odds-ratio*-st (log(OR) ehk shanside suhe)

$$\log(OR) = \text{logit}(p_1) - \text{logit}(p_2)$$

Odds-ratio

Kui meil on 2 katsetingimust (ravim/platseebo) ning 2 väljundit (näit elus/surnud), siis

- a - ravim/elus juhutude arv,
- b - ravim/surnud juhutude arv,
- c - platseebo/elus juhutude arv,
- d - platseebo/surnud juhutude arv.

$$OR = \frac{a/b}{c/d}$$

- $OR = 1$ Katsetingimus ei mõjuta väljundi *odds*-e
- $OR > 1$ Katsetingimus tõstab väljundi *odds*-e
- $OR < 1$ Katsetingimus langetab väljundi *odds*-e

Logistiline regressioon üldistab OR-i kaugemale kahest binaarsest muutujast. Kui meil on binaarne y-muutuja ja binaarne x-muutuja (X_1), pluss rida teisi x-muutujaid ($X_2 \dots X_n$), siis mitmese logistilise regressiooni X_1 -e tõusukoeffitsient β_1 on seotud tingimusliku OR-ga. $\exp(\beta_1)$ annab Y ja X vahelise OR-i, tingimisel, et teiste X-muutujate väärused on fikseeritud (see on tavaline sõltumatute muutujatega lineaarse regressiooni beta-koeffitsientide tõlgendamise tingimus).

OR-i kui suhtelise efekti suuruse tõlgendamine sõltub sündmuse $y = 1$ baastõenäosusest. Näiteks kui surm põhjusel x on tavapäraselt väga haruldane ja mingi keskkonnamõju annab $OR = 10$, siis tegelik tõus suremuses (surma tõenäosus keskkonnamõju tingimustes) võib olla tühine.

22.4.1.1 Logistiline regressioon brms-s

```
library(rethinking)
```

```

data(chimpanzees)
head(chimpanzees)
#>   actor recipient condition block trial prosoc_left chose_prosoc
#> 1      1          NA        0     1      2            0            1
#> 2      1          NA        0     1      4            0            0
#> 3      1          NA        0     1      6            1            0
#> 4      1          NA        0     1      8            0            1
#> 5      1          NA        0     1     10            1            1
#> 6      1          NA        0     1     12            1            1
#>   pulled_left
#> 1          0
#> 2          1
#> 3          0
#> 4          0
#> 5          1
#> 6          1

```

Intercept-i mudel ilma tõusuta

```

m_logreg_1 <- brm(data = chimpanzees,
                    family = binomial,
                    pulled_left ~ 1,
                    prior(normal(0, 10), class = Intercept))
write_rds(m_logreg_1, path= "data/m_logreg_1.fit")

tidy(m_logreg_1)
#>           term estimate std.error    lower    upper
#> 1 b_Intercept  0.323    0.0882  0.181   0.464
#> 2 lp__         -346.669   0.6652 -347.959 -346.194

```

Tõenäosus, et shimpans “pulled left”:

```

inv_logit_scaled(fixef(m_logreg_1))
#>           Estimate Est.Error Q2.5 Q97.5
#> Intercept      0.58     0.522 0.539 0.621

```

See tõenäosus peaks seega jääma kuhugi 54% ja 62% vaheline.

Nüüd ehtne ennustav logistiline regressioonimudel

```

m_logreg_2 <-
  brm(data = d, family = binomial,
       pulled_left ~ 1 + prosoc_left,
       prior = c(prior(normal(0, 10), class = Intercept),
                 prior(normal(0, 10), class = b)))
write_rds(m_logreg_2, path= "data/m_logreg_2.fit")

```

```
tidy(m_logreg_2)
#>   term  estimate std.error    lower    upper
#> 1 b_Intercept  0.0456    0.128  -0.165  0.259
#> 2 b_prosoc_left  0.5659    0.187   0.260  0.879
#> 3 lp__ -345.7102   1.014 -347.639 -344.741

exp(0.57)
#> [1] 1.77
```

Proportsionaalne odds = 1.76 kujutab endast suhet kahest tõenäosusest - et sündmus toimub ($y=1$) ja et sündmus ei toimu ($y=0$).

Kui prediktori prosoc_left väärthus muutumine 0-st 1-ks tõstab $y=1$ sündmuse $log\text{-}odds$ -i 0.57 võrra, siis kasvab proportsionaalselt $\exp(0.57) = 1.76$ ja $odds$, et toimub sündmus pull left ehk $y=1$ kasvab 76%.

Tegelik tõenäosuse muutus sõltub ka interceptist (α) ja teistest prediktoritest. Logistiline regressioon erineb tavalisest lineaarsest regressioonist selle poolest, et see mudeldab igal juhul muutujate vahelisi interaktsioone. Näiteks kui α on piisavalt suur, et garanteerida sündmuse $y = 1$ toimumine, siis ei tähenda $odds$ -ide tõus 73% võrra suurt midagi. Oletame, et $\alpha = 4$. Siis on sündmuse tõenäosus, ignoreerides köike muud `inv_logit_scaled(4)` = 0.98. Lisades sinna beta hinnagu 0.57 saame: `inv_logit_scaled(4 + 0.57)` = 0.99. See vastab 1% erinevusele, ehkki on samas 73% kasv suhtelise $odds$ -i ühikutes.

```
inv_logit_scaled(0.04562136 + 0.56590225)
#> [1] 0.648
```

$\Pr(Y = 1 | X = 1) = 64\%$.

```
inv_logit_scaled(0.04562136)
#> [1] 0.511
```

kui prosoc_left = 0, on $\Pr(Y = 1 | X = 0) = 51\%$.

ROC kõver peegeldab seda tagasihoidlikku erinevust.

Kui meil on binaarne (0/1) Y-mmutuja, siis ilma igasuguse ennustusjõuta loll mudel annab õige ennustuse pooltel juhtudest. Kui palju on meie mudel parem sellisest loll-mudelist? Sellele küsimusele annab granuleeritud vastuse roc kurv ja selle põhjal arvutatud *area under the curve* ehk auc.

Kõigepealt valmistame nn *confusion matriksi*. Selleks ennustame mudeli põhjal igale vaatlusele vastava pulled-left ehk $Y = 1$ tõenäosuse ja nende vaatluste puhul, mille $\Pr(Y=1) > 0.5$, loeme ennustustatukks, et simpans tömbab hooba vasakule. Siin on meie ennustuse *cut-off* 50%, aga see võiks olla ka teistsugune. Sõltuvalt sellest, kui palju me kardame eksida ühele või teisele poole (ennustada ekslikult, et $Y=1$ või magada maha tegelikke $Y=1$ sündmusi) peaksime iga kord valima oma hirme maksimaalselt maandava *cut-off*-i.

Kuna me teame, kuhu poole iga ahv igas katses tegelikult hooba tõmbas, on meil võimalik arvutada nii 1. tüüpi vigade (me ennustasime $Y=1$, aga tegelikult $Y=0$), kui ka vale-negatiivsete e 2. tüüpi vigade (me ennustasime $Y = 0$, aga tegelikult $Y=1$) sagedust. Loomulikult saame vigade sagedused arvutada kõikvõimalikele *cut-off*-idele.

```
glm.probs <- predict(m_logreg_2, type = "response") %>% as.data.frame()
#> Warning: Using 'binomial' families without specifying 'trials' on the left-
#> hand side of the model formula is deprecated.
glm.probs <- glm.probs[,1]
logit.pred <- factor(glm.probs > .5, levels=c(FALSE, TRUE),
                      labels=c("right", "left"))
table(chimpanzees$pulled_left, logit.pred, dnn=c("Actual", "Predicted"))
#>           Predicted
#> Actual right left
#>      0     8 204
#>      1    14 278
```

Siin, 0.5-se *cut-off*-iga tõenäosuse tabelis on meil 204 ahvi tõmmanud vasakule, samas kui mudel on ennustanud tõmmet paremale. Lisaks on 279 ahvi tõmmanud vasakule kooskõlas mudeli ennustusele, 8 ahvi on tõmmanud paremale kooskõlas mudeli ennustusega ja 13 ahvi on tõmmanud paremale hoolimata mudeli vastupidisest ennustusest.

1. tüüpi vigade sagedus on seega $204/(204+279)=0.42$ ja

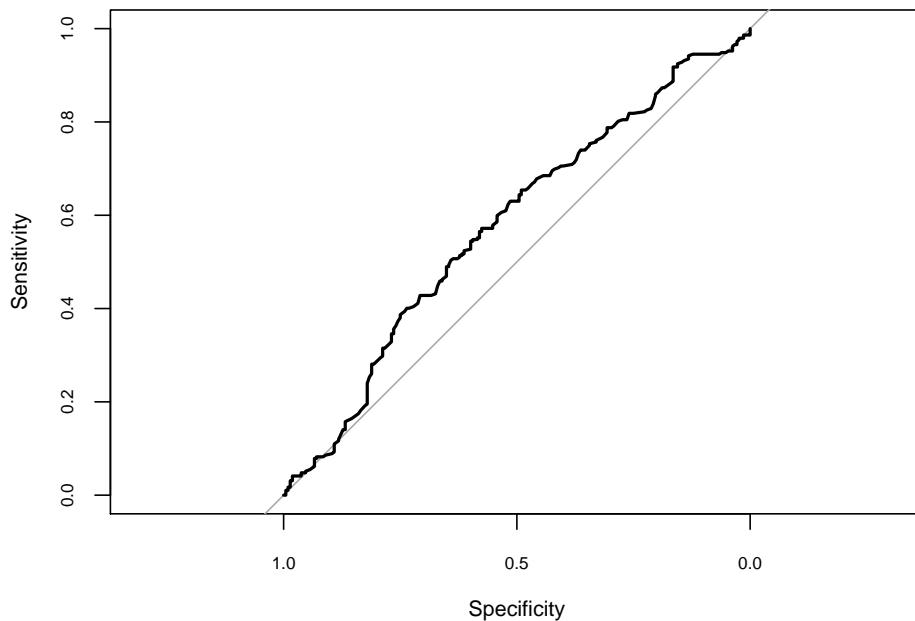
2. tüüpi vigade sagedus on $13/(8+13) = 0.61$.

Seega on sellel *cut-off*-il sensitiivsus $1 - 0.61 = 0.39$ ja spetsiifilisus $1 - 0.42 = 0.58$.

Klassifitseerimise täpsus *cut-off*-il 0.5 on $(8+279)/(8+204+13+279) = 0.57$

Roc kurv plotib mudeli ennususte sensitiivsuse ja spetsiifilisuse kõigil *cut-off* väärustustel.

```
library(pROC)
roccurve <- roc(chimpanzees$pulled_left ~ glm.probs)
plot(roccurve, cex.axis = 0.7, cex.lab = 0.8)
```



Nagu näha, kui spetsiifilisus on kõrge, siis sensitiivsus on madal ja vastupidi. See on alati nii. Fakt, et kurv ei kaugene kuigi palju keskjoonest, mis tähistab loll-mudeli ennustuse 50%-st tabavust, näitab et meie ennustuste kvaliteet on tagasihoidlik. *Area under the curve* ehk *auc* ütleb, et ennustuse tabavus on 59%.

```
auc(roccurve)
#> Area under the curve: 0.577
```

Sarnase mudeli saab fittida ka siis, kui $n > 1$ ja meil on igale ahvile countide suhted nr of pull-left/total pulls. Nüüd on meil vaja lisada trials(), kuhu läheb n kas ühe numbrina või muutujana, mis indekseerib sündmuste arvu ehk n-i. Antud juhul on kõikidel ahvidel katsete arv (n) 18.

```
chimp_aggr <- select(chimpanzees, actor, condition, prosoc_left, pulled_left) %>%
  group_by(actor, condition, prosoc_left) %>%
  summarise_at("pulled_left", sum)
m_logreg_3 <- brm(pulled_left | trials(18) ~ 1 + prosoc_left,
                    data = chimp_aggr,
                    family = binomial)
```

Koefitsendid tulevad samad, mis eelmisel mudelil.

Näide agregeeritud binoomsetele andmetele.

```
data(UCBadmit)
head(UCBadmit)
#>   dept applicant.gender admit reject applications
#> 1      A             male    512     313        825
```

```

#> 2   A      female    89     19      108
#> 3   B      male     353     207     560
#> 4   B      female    17      8       25
#> 5   C      male     120     205     325
#> 6   C      female   202     391     593

ucbadmit <- mutate(UCBadmit, male = case_when(
  applicant.gender == "male" ~ 1,
  TRUE ~ 0
))

m_ucadmit1 <- brm(admit | trials(applications) ~ 1 + male,
                     data = ubcadmit,
                     family = binomial,
                     prior = c(prior(normal(0, 10), class = Intercept),
                               prior(normal(0, 10), class = b)))
write_rds(m_ucadmit1, path = "data/m_ucadmit1.fit")

m_ucadmit1 <- read_rds("data/m_ucadmit1.fit")

tidy(m_ucadmit1)
#>           term estimate std.error    lower    upper
#> 1 b_Intercept -0.831    0.0509 -0.918 -0.748
#> 2   b_male     0.612    0.0626  0.512  0.718
#> 3     lp__ -433.695   0.9484 -435.621 -432.776

exp(0.61)
#> [1] 1.84

```

Mehed saavad suhtelise 84% eelise ülikooli sissesaamisel.

```

inv_logit_scaled(-0.83 + 0.61)
#> [1] 0.445

```

Meeskandidaadi tõenäosus sisse saada on 44%.

```

inv_logit_scaled(-0.83)
#> [1] 0.304

```

Naiskandidaadi tõenäosus sisse saada on 30%.

Kui palju erinevad vastuvõtmise tõenäosused (usaldusintervallidega)?

```

post <- posterior_samples(m_ucadmit1)
post %>% mutate(p_admit_male = inv_logit_scaled(b_Intercept + b_male),
                  p_admit_female = inv_logit_scaled(b_Intercept),
                  diff_admit = p_admit_male - p_admit_female) %>%
  summarise(`2.5%` = quantile(diff_admit, probs = .025),
            `50%` = median(diff_admit),

```

```

`97.5%` = quantile(diff_admit, probs = .975))
#>   2.5% 50% 97.5%
#> 1 0.115 0.142 0.17

```

Mudeldame otse küsimust, mis on naiste ja meeste erinevus sissesaamisel. intercepi surume nulli, et saada eraldi hinnang igale departmendile:

```

m_ucadmit2 <- brm(admit | trials(applications) ~ 0 + dept + male,
                     data = ucbadmit,
                     family = binomial,
                     prior(normal(0, 10), class = b))
write_rds(m_ucadmit2, path = "data/m_ucadmit2.fit")

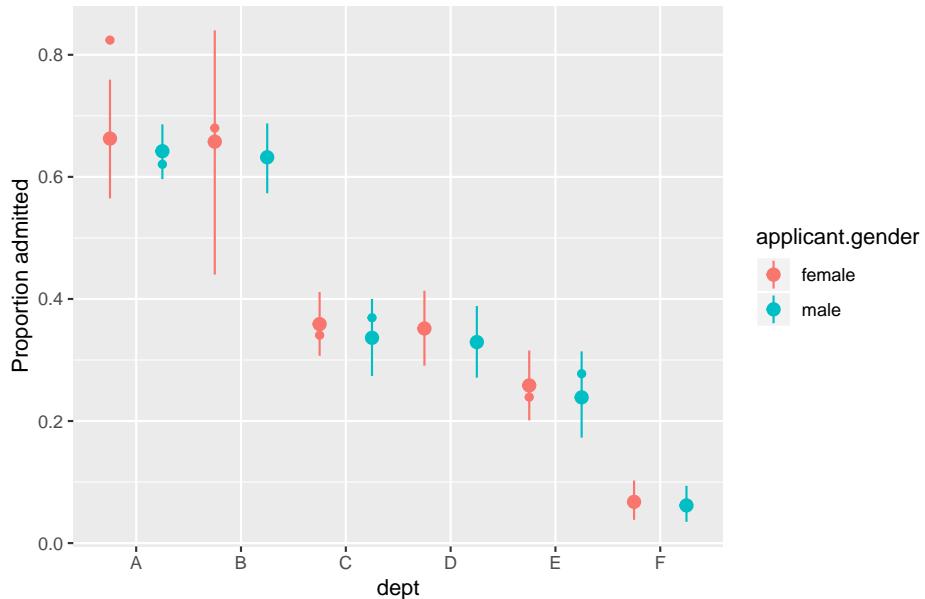
m_ucadmit2 <- read_rds("data/m_ucadmit2.fit")

tidy(m_ucadmit2)
#>   term estimate std.error   lower   upper
#> 1 b_deptA    0.684    0.0968   0.523   0.8426
#> 2 b_deptB    0.641    0.1157   0.453   0.8313
#> 3 b_deptC   -0.581    0.0743  -0.705  -0.4593
#> 4 b_deptD   -0.613    0.0841  -0.752  -0.4776
#> 5 b_deptE   -1.058    0.0974  -1.219  -0.9006
#> 6 b_deptF   -2.630    0.1556  -2.896  -2.3911
#> 7 b_male     -0.102    0.0807  -0.232   0.0368
#> 8 lp__     -70.630   1.8432 -74.018 -68.2572

pd <- position_dodge(1)
predict(m_ucadmit2) %>%
  as_tibble() %>%
  bind_cols(ucbadmit) %>%
  ggplot(aes(x = dept, y = admit / applications)) +
  geom_pointrange(aes(y = Estimate / applications,
                       ymin = Q2.5 / applications,
                       ymax = Q97.5 / applications,
                       color = applicant.gender), position = pd) +
  geom_point(aes(color = applicant.gender), position = pd) +
  labs(y = "Proportion admitted",
       title = "Posterior validation check")

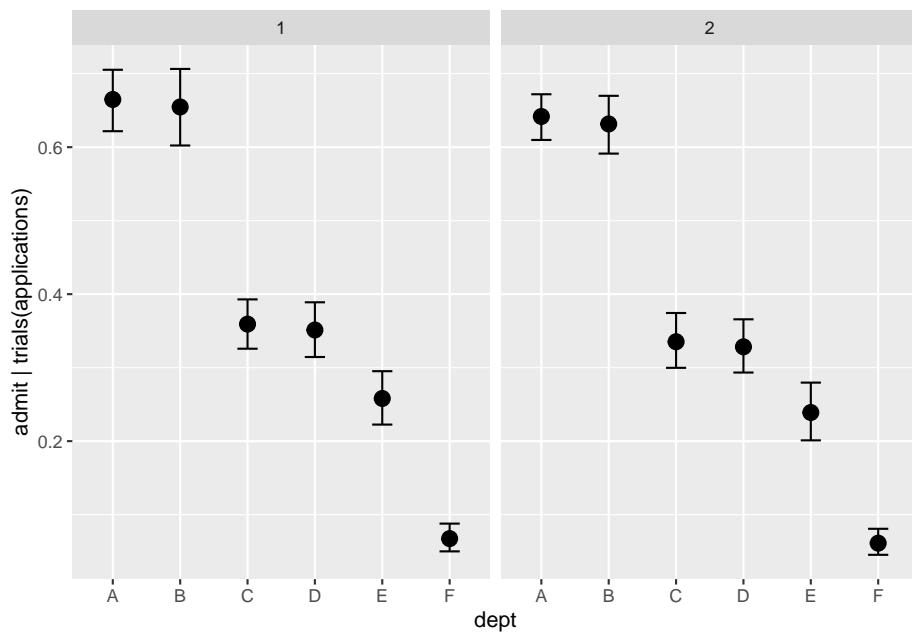
```

Posterior validation check



Ohho, kui vaadata deparmente eraldi, pole mingit kinnitust, et meestel oleks paremad võimalused ülikooli sisse saada.

```
marginal_effects(m_ucadmit2, effects ="dept", conditions = data.frame(male = c(0, 1)))
```



22.4.2 Y-muutujal 3+ kategoorialist väärustust

Building a generalized linear model from a multinomial likelihood is complicated, because as the event types multiply, so too do your modeling choices. And there are two different approaches to constructing the likelihoods, as well. The First is based directly on the multinomial likelihood and uses a generalization of the logit link.

When more than two types of unordered events are possible, and the probability of each type of event is constant across trials, then the maximum entropy distribution is the multinomial distribution. The conventional and natural link is this context is the multinomial logit. This link function takes a vector of scores, one for each K event types, and computed the probability of a particular type of event K.

Estimate the association between person's family income and which career (there are 3 choices) he chooses.

```
N <- 100
set.seed(2078)
# simulate family incomes for each individual
family_income <- runif(N)
# assign a unique coefficient for each type of event
b <- (1:-1)
career <- rep(NA, N) # empty vector of choices for each individual

for (i in 1:N) {
  score <- 0.5 * (1:3) + b * family_income[i]
  p <- softmax(score[1], score[2], score[3])
  career[i] <- sample(1:3, size = 1, prob = p)
}

mult_logistic_m1 <-
  brm(data = list(career = career, family_income = family_income),
      family = categorical(link = "logit"),
      career ~ 1 + family_income)

write_rds(mult_logistic_m1, path = "data/mult_logistic_m1.fit")
```

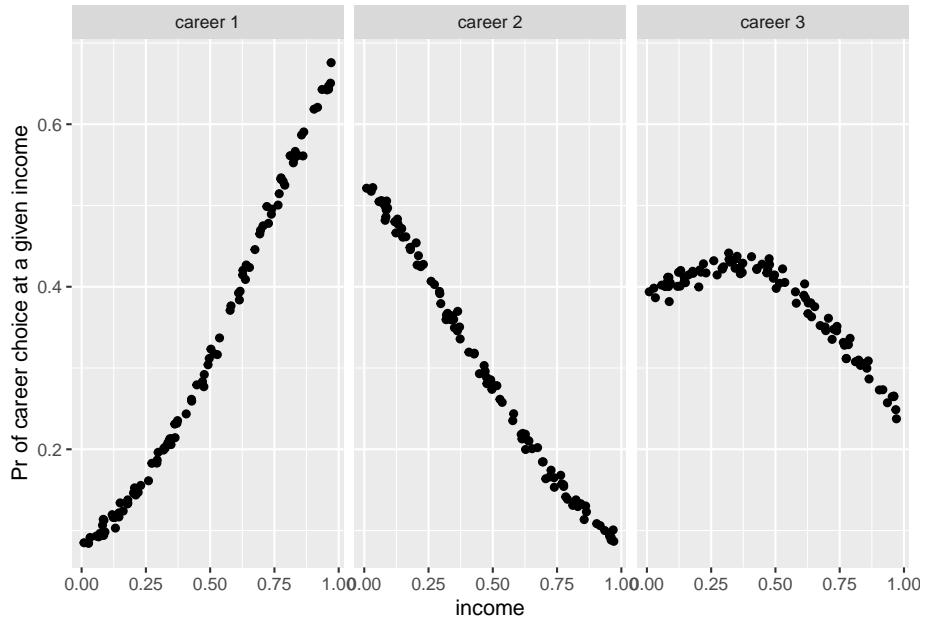
Parameetrid ei saa otse tõlgendada. Selle asemel on mõistlik töötada mudeli ennustuste tasemel konverteerides parameetrid 3ks tõenäosusteks, et inimene kindlal perekonna sissetuleku tasemel valib karjääri 1, 2 või 3.

```
pred1 <- predict(mult_logistic_m1) %>% as_tibble()
pred1$career <- career
pred1$income <- family_income
pred1_l <- reshape2::melt(pred1, id.vars = 4:5, measure.vars = 1:3)
pred1_l <- pred1_l %>%
```

```

mutate(variable = case_when(variable == "P(Y = 1)" ~ "career 1",
                            variable == "P(Y = 2)" ~ "career 2",
                            variable == "P(Y = 3)" ~ "career 3"))
ggplot(pred1_1, aes(income, value)) + geom_point() + facet_wrap(~variable) +
  ylab("Pr of career choice at a given income")

```



Y-muutujal on 3+ kategoorialist väärust, mis on ordinaalselt järjestatud

Kui näiteks Y-muutuja väärustused on “vähe”, “rohkem”, “palju rohkem”, siis me saame aru, et 3 väärustust on küll järjestatud, aga nende vahelised kaugused andmeruumis ei pruugi olla võrdsed. Nüüd kasutame mudelit vormis:

```

brm(y ~ x + (1|id),
  data=d,
  family=cumulative("logit"),
  threshold="flexible")

```

Kumulatiivse mudeliga ordinaalse logistilise regressiooni üks eeldustest (proportional odds assumption) on, et y-muutuja tasemete iga võimalik paar on sama tõusuga (identsed beta koefitsiendid, erinevad ainult interceptid). Tasub ka meeles pidada, et mudelisse X-muutujaid lisades võime selle eelduse pekki keerata.

22.4.3 zero inflated mudelid

Kasulikud siis, kui teil on Y-s rohkem nulle kui võiks arvata. Näiteks, kui proovida hinnata, mitu suitsu päevas tõmmatakse andmestikust, mis sisaldab mittesuitsetajaid.

selle mudeli spetsifitseerimiseks pole vaja teha muud, kui õelda brm() argument family = zero_inflated_poisson(), zero_inflated_beta(), _binomial(), _negbinomial().

zinb data: mitu kala turist püüab.

```
zinb <- read_csv("http://stats.idre.ucla.edu/stat/data/fish.csv")
zinb

brm(count ~ persons + child + camper,
    data = zinb, family = zero_inflated_poisson())
```

sellel mudelil on parameeter zi ehk zero inflated probability, mille väärthus annab infleeritud 0-de suhte köikidesse 0-desse,

järgmiseks püüame veel lisaks ennustada zi väärust lähtuvalt laste arvust (lastega pered võiks vähem kalastada)

```
brm(bf(count ~ persons + child + camper,
        zi ~ child), data = zinb, family = zero_inflated_poisson())
```

Nüüd on meil parameetrid zi_Intercept ja zi_child

22.4.4 additiivsed distributsioonilised mudelid

inkorporeerime splinid multitaseme mudelisse - siis kui me ei tea y ja x-i suhetekuju.

simuleerime andmed

```
dat_smooth <- mgcv::gamSim(eg = 6, n = 200, scale = 2, verbose = FALSE)
#> Gu & Wahba 4 term additive model

head(dat_smooth[, 1:6])
#>      y     x0     x1     x2     x3     f
#> 1  5.00  0.945  0.192  0.5494  0.246  7.69
#> 2  9.08  0.126  0.194  0.0512  0.963  9.04
#> 3 16.63  0.807  0.508  0.1144  0.396 17.35
#> 4 19.70  0.399  0.823  0.9513  0.748 19.09
#> 5  6.84  0.379  0.396  0.8617  0.660  7.34
#> 6 20.27  0.123  0.826  0.1846  0.197 20.15
```

x_0 kuni x_3 on prediktorid, fac on faktor veerg, mis indikeerib nested andmestruktuuri. Me ennustame y väärtsusi x_0 ja x_1 järgi, ja lisaks laseme residuaalse sd varieeruma x_0 smoothing termi ja fac gruppide interceptide järgi.

```
fit_smooth1 <- brm(
  bf(y ~ s(x1) + s(x2) + (1|fac), sigma ~ s(x0) + (1|fac)),
  data = dat_smooth, family = gaussian()
)
```

22.5 Monotoonilised efektid

näit prediktor muutuja: suitsetab palju - vähe - väga vähe. Me ei eelda, et 3 taset oleks üksteisest sama kaugel - ei modelleeri seda pideva muutujana.

```
fit1 <- brm(y ~ mo(x), data = d)
```

Nüüd tekivad meile Simplex parameetrid. Nende prior on kanooniliselt ühe parameetriga (alpha) Dirichlet prior, mis on beta jaotuse multivariaatne üldistus.

Siin me eeldame, et kõrvuti kategooriate erinevused on samad - e lähevad sama priori alla. Kui me eeldame, et meil on näit 3 järjestikust monotoonilist taset ja, et madalatel väärustustel (1) on monotoonilise muutuja mõju Y -le suurem, siis anname alphale kõrgema väärtsuse. Meie kõlemes näites on vaja spetsifitseerida vektor 3 alpha-ga.

```
prior <- prior(dirichlet(c(2, 1, 1)), class = "...", coef = "...")
fit4 <- brm(y ~ mo(x), data = d,
             prior = prior, sample_prior = TRUE)
```

monotooniliste muutujate interaktsiooni mudeldamine

```
fit5 <- brm(y ~ mo(x)*age, data = d)
marginal_effects(fit5, "x:age")
```

mitmetasemelised monotoonilised mudelid

```
fit6 <- brm(y ~ mo(x)*age + (mo(x) | city), data = d)
```

22.5.1 Multivariaatsed mudelid

Mitu y muutujat, millega igaühel on oma prediktorid.

Eurasian blue tit: predict the tarsus length as well as the back color of chicks. Half of the brood were put into another fosternest, while the other half stayed in the fosternest of their own dam. This allows to separate genetic from environmental factors. Additionally, we have information about the hatchdate and sex of the chicks (the latter being known for 94% of the animals).

```

data("BTdata", package = "MCMCglmm")
head(BTdata)

fit1 <- brm(
  cbind(tarsus, back) ~ sex + hatchdate + (1|p|fosternest) + (1|q|dam),
  data = BTdata, chains = 2, cores = 2
)
add_ic(fit1) <- "loo"
summary(fit1)
pp_check(fit1, resp = "tarsus")
pp_check(fit1, resp = "back")

```

The term $(1|p|fosternest)$ indicates a varying intercept over fosternest. By writing $|p|$ in between we indicate that all varying effects of fosternest should be modeled as correlated. This makes sense since we actually have two model parts, one for tarsus and one for back. The indicator p is arbitrary and can be replaced by other symbols that comes into your mind. Similarly, the term $(1|q|dam)$ indicates correlated varying effects of the genetic mother of the chicks.

The summary output of multivariate models closely resembles those of univariate models, except that the parameters now have the corresponding response variable as prefix. Within dams, tarsus length and back color seem to be negatively correlated, while within fosternests the opposite is true. This indicates differential effects of genetic and environmental factors on these two characteristics. Further, the small residual correlation `rescor(tarsus, back)` on the bottom of the output indicates that there is little unmodeled dependency between tarsus length and back color. Although not necessary at this point, we have already computed and stored the LOO information criterion of `fit1`, which we will use for model comparisons.

Me võime anda ette erinevad valemid kummagiile y-muutujale nii

```

bf_tarsus <- bf(tarsus ~ sex + (1|p|fosternest) + (1|q|dam))
bf_back <- bf(back ~ hatchdate + (1|p|fosternest) + (1|q|dam))
fit2 <- brm(bf_tarsus + bf_back, data = BTdata, chains = 2, cores = 2)

```

We change our model in various directions at the same time. Remember the slight left skewness of tarsus, which we will now model by using the `skew_normal` family instead of the `gaussian` family. Since we do not have a multivariate normal (or student-t) model, estimating residual correlations is no longer possible. We make this explicit using the `set_rescor` function. We investigate if the relationship of back and hatchdate is really linear as previously assumed by fitting a non-linear spline of hatchdate. On top of it, we model separate residual variances of tarsus for males and females chicks.

```

bf_tarsus <- bf(tarsus ~ sex + (1|p|fosternest) + (1|q|dam)) +
  lf(sigma ~ 0 + sex) + skew_normal()
bf_back <- bf(back ~ s(hatchdate) + (1|p|fosternest) + (1|q|dam)) +

```

```
gaussian()
fit3 <- brm(bf_tarsus + bf_back + set_rescor(FALSE), data = BTdata)
```

22.5.2 Mittelineaarsed mudelid

```
b <- c(2, 0.75)
x <- rnorm(100)
y <- rnorm(100, mean = b[1] * exp(b[2] * x))
dat1 <- data.frame(x, y)

prior1 <- prior(normal(1, 2), nlpar = "b1") +
  prior(normal(0, 2), nlpar = "b2")
fit1 <- brm(bf(y ~ b1 * exp(b2 * x), b1 + b2 ~ 1, nl = TRUE),
            data = dat1, prior = prior1)
```

Siin on iga mittelineaarne parameeter (b_1 ja b_2) eraldi modelleeritud ~ 1 abil. Argument $b_1 + b_2 \sim 1$ (lühivorm: $b_1 \sim 1$, $b_2 \sim 1$) ütleb, mis muutujad valemist on parameetrid, mille väärust tuleb hinnata. Lisaks spetsifitseerib see igale parameetrile lineaarsed prediktorid. Mitte-lineaarsed parameetrid on tegelikult kohahoidjad lineaarsetele prediktor-termidele. Kuna me siin ei ennusta b_1 ja b_2 lisaparameetrite kaudu, on meil intercept-only mudel kummagile.

Priors on population-level parameters (i.e., ‘fixed effects’) are often mandatory to identify a non-linear model. Thus, brms requires the user to explicitly specify these priors. In the present example, we used a $\text{normal}(1, 2)$ prior on (the population-level intercept of) b_1 , while we used a $\text{normal}(0, 2)$ prior on (the population-level intercept of) b_2 . Setting priors is a non-trivial task in all kinds of models, especially in non-linear models, so you should always invest some time to think of appropriate priors. Quite often, you may be forced to change your priors after fitting a non-linear model for the first time, when you observe different MCMC chains converging to different posterior regions. This is a clear sign of an identification problem and one solution is to set stronger (i.e., more narrow) priors.

Chapter 23

brms süntaks

Symbol	Example	Meaning
+	+ x	include this variable
-	-	x delete this variable
:	x : z	include the interaction between these variables
*	x * z	include these variables and the interactions between them
/	x / z	nesting: include z nested within x
	x z	conditioning: include x given z
^	(u + v + w + z)^3	include these variables and all interactions up to three way
poly	poly(x,3)	polynomial regression: orthogonal polynomials
Error	Error(a/b)	specify an error term
I	I(x*z)	as is: include a new variable consisting of these variables multiplied. (x^2) means
1	- 1	intercept: delete the intercept (regress through the origin)

üldine vorm:

`response ~ pterms + (gterms | group)`

kasuta g1:g2 või g1/g2, kui nii g1 kui g2 on sobilikud grupeerivad faktorid.

operaator loob uue grupeeriva faktori, mis kombineerib g1 ja g2 tasemed.

/ operaator viitab nested struktuurile (kool - koolitüüp)

(1 | g1/g2), tähendab tegelikult $(1 | g1) + (1 | g1:g2)$.

$(1 | g1 + g2)$ on sama, mis $(1 | g1) + (1 | g2)$.

|| kasutades $(x || g1)$ ei modelleeri me grupi-taseme korrelatsioone. See on hea, kui mudeli fittimine muidu ei tööta.

kuidas mudeldada sama grupeeriva faktori korrelatsioone üle mitme regresioonivõrandi? Selleks laiendame | operaatori ||, kus on suvaline väärthus. Näiteks kui termid $(x1|ID|g1)$ and $(x2|ID|g1)$ esinevad kuskil samas või erinevates võrandites, modelleeritakse need kui korreleeritud.

alternatiivseid grupeeivaid struktuure saab väljendada nii:

`(gterms | fun(group)).`

Hetkel on meil 2 sellist fun-i: `gr()` annab default käitumise ja `mm()` annab multi-membership termid. Näiteks `brm(y ~ 1 + (1 | mm(s1, s2))` modelleerib seda, kuidas lapsed võivad õppida kahes koolis (s1 ja s2) eri aegadel

`gr()` lisatakse muidu automaatselt, aga seda spetsifitseerides saab kirjutada `y ~ x + (1|gr(g1, by = g2))`, mis tähendab, et grupeeriva muutja g1 sees lahitatakse veel gruppidesse g2 muutuja tasemete järgi - ja iga g2 grupp modelleeritakse iseseisvalt (ilma shrinkageta)

Mittelineaarne mudel:

`y = b1(1 - exp(-(x/b2)**b3)`

y ja x seos parameetritega b1..b3 Oletame, et tahame kõik parameetrid fittida grupeeriva muutuja g tasemete järgi ja et grupeeriva muutuja tasemel efektid oleks omavahel korreleeritud. Lisaks ennustame me b1-e kovariaat z järgi. See kõik läheb järgmisesse võrrandisüsteemi kus, lisaks mitte-lineaarsele võrrandile, igale parameetrile b1...b3 vastab oma lineaarne võrrand:

`y ~ b1 * (1 - exp(-(x / b2) ^ b3) b1 ~ z + (1|ID|g) b2 ~ (1|ID|g)`
`b3 ~ (1|ID|g)`

lisaks on mudeli keeles silumistermid s() ehk spline ja t2() ehk bivariate tensor spline, mis tulevad mgcv paketist. Näiteks `rentsqm ~ t2(area, yearc) + (1|district)`

category specific effects cs,

monotonic effects mo,

noise-free effects me, or

Gaussian process terms gp.

additional information on the response variable may be specified via `response | aterms ~ <predictor terms>`. The aterms part may contain multiple terms of the form `fun()` separated by + each providing special information on the response variable. This allows among others to weight observations, provide known standard errors for meta-analysis, or model censored or truncated data.

23.1 General formula structure

`response | aterms ~ pterms + (gterms | group)`

The pterms - effects that are assumed to be the same across observations ('population-level' effects).

gterms - effects that are assumed to vary across grouping variables specified in group ('group-level' effects).

23.1.1 Group-level terms: $x||g$, $1|ID|g$, $gr(x, by =)$, $mm(g1, g2)$

Multiple grouping factors each with multiple group-level effects are possible.

Use $||$ in grouping terms to prevent correlations from being modeled.

It is possible to model different group-level terms of the same grouping factor as correlated (even across different formulas, e.g., in non-linear models) by using $||$ instead of $|$. All group-level terms sharing the same ID will be modeled as correlated. for instance, the terms $(1+x|2|g)$ and $(1+z|2|g)$ mean that correlations between x and z will be estimated. correlated group-level effects across parameters: `bf(y ~ a1 - a2^x, a1 ~ 1 + (1|2|g), a2 ~ x + (x|2|g), n1 = TRUE)`

If levels of the grouping factor belong to different sub-populations, it may be reasonable to assume a different covariance matrix for each of the sub-populations. For instance, the variation within the treatment group and within the control group in a randomized control trial might differ. Suppose that y is the outcome, and x is the factor indicating the treatment and control group. Then, we could estimate different hyper-parameters of the varying effects (in this case a varying intercept) for treatment and control group via $y \sim x + (1 | gr(subject, by = x))$.

A multi-membership term with two members: $(1 | mm(g1, g2))$, where $g1$ and $g2$ specify the first and second member, respectively. If x varies across the levels of $g1$ and $g2$, we can save the respective x values in the variables $x1$ and $x2$ and then model the varying effect as $(1 + mmc(x1, x2) | mm(g1, g2))$.

multilevel w smoothing terms `brmsformula(y ~ x1*x2 + s(z) + (1+x1|1) + (1|g2))`

additionally predict 'sigma' `brmsformula(y ~ x1*x2 + s(z) + (1+x1|1) + (1|g2), sigma ~ x1 + (1|g2))`

23.1.2 Special predictor terms (`s()`, `t2()`, `gp()`, `mo()`, `me()`, `mi()`, `cs()`)

Smoothing terms are modeled using `s()` and `t2()` in the pterms part of the formula.

Gaussian process terms can be fitted using `gp()` in the pterms. Similar to smooth terms, Gaussian processes can be used to model complex non-linear relationships, for instance temporal or spatial autocorrelation. However, they

are computationally demanding and are thus not recommended for very large datasets.

The pterms and gterms parts may contain non-standard effect types: monotonic mo(predictor), measurement error me(predictor, sd_predictor), missing value mi(predictor), and category specific effects cs().

Category specific effects can only be estimated in ordinal models and are explained in more detail in the package’s main vignette (type vignette(“brms_overview”)).

A monotonic predictor must either be integer valued or an ordered factor. Predictor categories (or integers) are not assumed to be equidistant with respect to their effect on the response variable. The distance between adjacent categories is estimated from the data and may vary across categories. One parameter takes care of the direction and size of the effect similar to an ordinary regression parameter, while an additional parameter vector estimates the normalized distances between consecutive predictor categories.

Often predictors contain measurement error. Effects of noise-free predictors can be modeled using the me (for ‘measurement error’) function. If x is a measured predictor with known measurement error sdx, we can include it via $y \sim me(x, sdx)$. If x2 is another measured predictor with corresponding error sdx2 and z is a predictor without error (e.g., an experimental setting), we can model all main effects and interactions: $y = me(x, sdx) * me(x2, sdx2) * z$.

Using a variable with missing values as predictors requires two things, First, we need to specify that the predictor contains missings that should to be imputed. If x is a predictor with missings and z is a predictor without missings, we go for $y \sim mi(x) + z$. Second, we need to model x as an additional response with corresponding predictors and the addition term mi(). In our example, we could write $x \mid mi() \sim z$.

```
bf(bmi ~ age * mi(chl)) + bf(chl | mi() ~ age) + set_rescor(FALSE).
```

23.1.3 Additional response information

aterms part may contain multiple terms of the form fun() separated by + each providing special information on the response variable. fun can be replaced with either se, weights, cens, trunc, trials, cat, or dec.

For families gaussian, student and skew_normal, it is possible to specify standard errors of the observations, thus allowing to perform meta-analysis. Suppose that the variable y_i contains the effect sizes from the studies and se_i the corresponding standard errors. Then, fixed and random effects meta-analyses can be conducted using the formulas $y_i \mid se(se_i) \sim 1$ and $y_i \mid se(se_i) \sim 1 + (1|study)$, respectively, where study is a variable uniquely identifying every study.

Meta-regression can be performed via `y_i | se(se_i) ~ 1 + mod1 + mod2 + (1|study)` or `y_i | se(se_i) ~ 1 + mod1 + mod2 + (1 + mod1 + mod2|study)`, where mod1 and mod2 represent moderator variables. By default, the standard errors replace the parameter sigma. To model sigma in addition to the known standard errors, set argument sigma in function se to TRUE, for instance, `y_i | se(se_i, sigma = TRUE) ~ 1`.

For all families, weighted regression may be performed using weights in the aterms part. Internally, this is implemented by multiplying the log-posterior values of each observation by their corresponding weights. Suppose that variable `we_i` contains the weights and that `yi` is the response variable. Then, formula `y_i | weights(we_i) ~ predictors` implements a weighted regression.

With the exception of categorical, ordinal, and mixture families, left, right, and interval censoring can be modeled through `y | cens(censored) ~ predictors`. The censoring variable (named censored in this example) should contain the values ‘left’, ‘none’, ‘right’, and ‘interval’ (or equivalently -1, 0, 1, and 2) to indicate that the corresponding observation is left censored, not censored, right censored, or interval censored. For interval censored data, a second variable (let’s call it `y2`) has to be passed to `cens`. In this case, the formula has the structure `y | cens(censored, y2) ~ predictors`. While the lower bounds are given in `y`, the upper bounds are given in `y2` for interval censored data. Intervals are assumed to be open on the left and closed on the right: $(y, y2]$.

With the exception of categorical, ordinal, and mixture families, the response distribution can be truncated using the `trunc` function in the addition part. If the response variable is truncated between, say, 0 and 100, we can specify this via `yi | trunc(lb = 0, ub = 100) ~ predictors`. Instead of numbers, variables in the data set can also be passed allowing for varying truncation points across observations. Defining only one of the two arguments in `trunc` leads to one-sided truncation.

For all continuous families, missing values in the responses can be imputed within Stan by using the addition term `mi`. This is mostly useful in combination with `mi` predictor terms as explained above under ‘Special predictor terms’.

For families `binomial` and `zero_inflated_binomial`, addition should contain a variable indicating the number of trials underlying each observation: `success | trials(n)`. If the number of trials is constant across all observations, say 10, we may also write `success | trials(10)`.

For all ordinal families, aterms may contain a term `cat(number)` to specify the number of categories (e.g, `cat(7)`). If not given, the number of categories is calculated from the data.

Multiple addition terms may be specified at the same time using the `+` operator, for instance `formula = y_i | se(se_i) + cens(censored) ~ 1` for a censored meta-analytic model.

23.1.4 Formula syntax for non-linear models

The non-linear model can be specified within the formula argument. Suppose, that we want a non-linear model being defined via formula = $y \sim \alpha - \beta * \lambda x$. To tell brms that this is a non-linear model, argument nl = TRUE. Now we have to specify a model for each of the non-linear parameters. Let's say we just want to estimate those three parameters with no further covariates or random effects. Then we can pass alpha + beta + lambda ~ 1 or equivalently (and more flexible) alpha ~ 1, beta ~ 1, lambda ~ 1 to the ... argument. If we have another predictor z and observations nested within the grouping factor g, we may write for instance alpha ~ 1, beta ~ 1 + z + (1|g), lambda ~ 1. We are using z and g only for the prediction of beta, but we might also use them for the other non-linear parameters.

Non-linear models may not be uniquely identified and / or show bad convergence. For this reason it is mandatory to specify priors on the non-linear parameters.

```
bf(y ~ a1 - a2^x, a1 + a2 ~ 1, nl = TRUE) simple nl model
```

23.1.5 Formula syntax for predicting distributional parameters

It is also possible to predict parameters of the response distribution such as the residual standard deviation sigma in gaussian models or the hurdle probability hu in hurdle models. The syntax closely resembles that of a non-linear parameter, for instance sigma ~ x + s(z) + (1+x|g).

Alternatively, one may fix distributional parameters to certain values. This is useful when models become too complicated and otherwise have convergence issues. The quantile parameter of the asym_laplace distribution is a good example where it is useful. By fixing quantile, one can perform quantile regression for the specified quantile. For instance, quantile = 0.25 allows predicting the 25%-quantile. Furthermore, the bias parameter in drift-diffusion models, is assumed to be 0.5 (i.e. no bias) in many applications. To achieve this, simply write bias = 0.5. Other possible applications are the Cauchy distribution as a special case of the Student-t distribution with nu = 1, or the geometric distribution as a special case of the negative binomial distribution with shape = 1. Furthermore, the parameter disc ('discrimination') in ordinal models is fixed to 1 by default and not estimated, but may be modeled as any other distributional parameter if desired ('disc' can only be positive, which is achieved by applying the log-link).

In categorical models, distributional parameters do not have fixed names. Instead, they are named after the response categories (excluding the first one, which serves as the reference category), with the prefix 'mu'. If, for instance, categories are named cat1, cat2, and cat3, the distributional parameters will be named mucat2 and mucat3.

Some distributional parameters currently supported by `brmsformula` have to be positive (a negative standard deviation or precision parameter does not make any sense) or are bounded between 0 and 1 (for zero-inflated / hurdle probabilities, quantiles, or the initial bias parameter of drift-diffusion models). However, linear predictors can be positive or negative, and thus the log link (for positive parameters) or logit link (for probability parameters) are used by default to ensure that distributional parameters are within their valid intervals. This implies that, by default, effects for such distributional parameters are estimated on the log / logit scale and one has to apply the inverse link function to get to the effects on the original scale. Alternatively, it is possible to use the identity link to predict parameters on their original scale, directly. However, this is much more likely to lead to problems in the model fitting, if the parameter actually has a restricted range.

23.1.6 Formula syntax for mixture models

If not specified otherwise, all mean parameters of the mixture components are predicted using the right-hand side of formula.

distributional parameters of mixture distributions have the same name as those of the corresponding ordinary distributions, but with a number at the end to indicate the mixture component. For instance, if you use family `mixture(gaussian, gaussian)`, the distributional parameters are `sigma1` and `sigma2`. distributional parameters of the same class can be fixed to the same value. For the above example, we could write `sigma2 = "sigma1"` to make sure that both components have the same residual standard deviation, which is in turn estimated from the data.

```
specify different predictors for different mixture components mix <- mixture(gaussian, gaussian) bf(y ~ 1, mu1 ~ x, mu2 ~ z, family = mix)
```

```
fix both residual standard deviations to the same value bf(y ~ x, sigma2 = "sigma1", family = mix)
```

In addition, there are two types of special distributional parameters. The first are named `mu`, that allow for modeling different predictors for the mean parameters of different mixture components. For instance, if you want to predict the mean of the first component using predictor `x` and the mean of the second component using predictor `z`, you can write `mu1 ~ x` as well as `mu2 ~ z`. The second are named `theta`, which constitute the mixing proportions. If the mixing proportions are fixed to certain values, they are internally normalized to form a probability vector. If one seeks to predict the mixing proportions, all but one of them has to be predicted, while the remaining one is used as the reference category to identify the model. The softmax function is applied on the linear predictor terms to form a probability vector.

23.1.7 Formula syntax for multivariate models

Multivariate models may be specified using cbind notation or with help of the mvbf function. Suppose that y_1 and y_2 are response variables and x is a predictor. Then $\text{cbind}(y_1, y_2) \sim x$ specifies a multivariate model. The effects of all terms specified at the RHS of the formula are assumed to vary across response variables. For instance, two parameters will be estimated for x , one for the effect on y_1 and another for the effect on y_2 . This is also true for group-level effects. When writing, for instance, $\text{cbind}(y_1, y_2) \sim x + (1+x|g)$, group-level effects will be estimated separately for each response. To model these effects as correlated across responses, use the ID syntax (see above). For the present example, this would look as follows: $\text{cbind}(y_1, y_2) \sim x + (1+x|2|g)$. Of course, you could also use any value other than 2 as ID.

It is also possible to specify different formulas for different responses. If, for instance, y_1 should be predicted by x and y_2 should be predicted by z , we could write $\text{mvbf}(y_1 \sim x, y_2 \sim z)$. Alternatively, multiple brmsformula objects can be added to specify a joint multivariate model (see ‘Examples’).

23.2 brms likelihoods

A character string naming the distribution of the response variable.

```
student(link = "identity", link_sigma = "log", link_nu = "logm1")
bernoulli(link = "logit")
negbinomial(link = "log", link_shape = "log")
geometric(link = "log")
lognormal(link = "identity", link_sigma = "log")
shifted_lognormal(link = "identity", link_sigma = "log", link_ndt = "log")
skew_normal(link = "identity", link_sigma = "log", link_alpha = "identity")
exponential(link = "log")
weibull(link = "log", link_shape = "log")
exgaussian(link = "identity", link_sigma = "log", link_beta = "log")
Beta(link = "logit", link_phi = "log")
hurdle_poisson(link = "log")
hurdle_negbinomial(link = "log", link_shape = "log", link_hu = "logit")
hurdle_gamma(link = "log", link_shape = "log", link_hu = "logit")
hurdle_lognormal(link = "identity", link_sigma = "log", link_hu = "logit")
```

```

zero_inflated_beta(link = "logit", link_phi = "log", link_zi = "logit")
zero_one_inflated_beta(link = "logit", link_phi = "log", link_zoi = "logit",
link_coi = "logit")
zero_inflated_poisson(link = "log", link_zi = "logit")
zero_inflated_negbinomial(link = "log", link_shape = "log", link_zi = "logit")
zero_inflated_binomial(link = "logit", link_zi = "logit")
categorical(link = "logit")
cumulative(link = "logit", link_disc = "log", threshold = c("flexible", "equidistant"))

```

- Family gaussian with identity link leads to linear regression.
- Family student with identity link leads to robust linear regression that is less influenced by outliers.
- Family skew_normal can handle skewed responses in linear regression.
- Families poisson, negbinomial, and geometric with log link lead to regression models for count data.
- Families binomial and bernoulli with logit link leads to logistic regression and family categorical to multi-logistic regression when there are more than two possible outcomes.
- Families cumulative, cratio ('continuation ratio'), sratio ('stopping ratio'), and acat ('adjacent category') leads to ordinal regression.
- Families Gamma, weibull, exponential, lognormal, frechet, and inverse.gaussian can be used (among others) for survival regression.
- Families weibull, frechet, and gen_extreme_value ('generalized extreme value') allow for modeling extremes.
- Family asym_laplace allows for quantile regression when fixing the auxiliary quantile parameter to the quantile of interest.
- Family exgaussian ('exponentially modified Gaussian') and shifted_lognormal are especially suited to model reaction times.
- The wiener family provides an implementation of the Wiener diffusion model. For this family, the main formula predicts the drift parameter 'delta' and all other parameters are modeled as auxiliary parameters (see brmsformula for details).
- Families hurdle_poisson, hurdle_negbinomial, hurdle_gamma, hurdle_lognormal, zero_inflated_poisson, zero_inflated_negbinomial, zero_inflated_binomial, zero_inflated_beta, and zero_one_inflated_beta allow to estimate zero-inflated and hurdle models. These models can be very helpful when there are many zeros in the data (or ones in case of

one-inflated models) that cannot be explained by the primary distribution of the response. Families hurdle_lognormal and hurdle_gamma are especially useful, as traditional lognormal or Gamma models cannot be reasonably fitted for data containing zeros in the response.

A list of all possible links for each family:

- The families gaussian, student, skew_normal, exgaussian, asym_laplace, and gen_extreme_value accept the links (as names) identity, log, and inverse;
- families poisson, negbinomial, geometric, zero_inflated_poisson, zero_inflated_negbinomial, hurdle_poisson, and hurdle_negbinomial the links log, identity, and sqrt;
- families binomial, bernoulli, Beta, zero_inflated_binomial, zero_inflated_beta, and zero_one_inflated_beta the links logit, probit, probit_approx, cloglog, cauchit, and identity;
- families cumulative, cratio, sratio, and acat the links logit, probit, probit_approx, cloglog, and cauchit; family categorical the link logit;
- families Gamma, weibull, exponential, frechet, and hurdle_gamma the links log, identity, and inverse; families lognormal and hurdle_lognormal the links identity and inverse;
- family inverse.gaussian the links $1/\mu^2$, inverse, identity and log; family von_mises the link tan_half; family wiener the link identity.

The first link mentioned for each family is the default.

23.3 priors

23.3.1 Population-level ('fixed') effects

If y is predicted by x_1 and x_2 then x_1 and x_2 have regression parameters b_{x1} and b_{x2} . The default prior for population-level effects (including monotonic and category specific effects) is an improper flat prior. Other common options are normal priors or student-t priors. If we want to have a normal prior with mean 0 and standard deviation 5 for x_1 , and a unit student-t prior with 10 degrees of freedom for x_2 , we can specify this via `set_prior("normal(0,5)", class = "b", coef = "x1")` and `set_prior("student_t(10,0,1)", class = "b", coef = "x2")`. To put the same prior on all population-level effects at once, `set_prior("", class = "b")`. This also leads to faster sampling, because priors can be vectorized in this case. Both ways of defining priors can be combined using for instance `set_prior("normal(0,2)", class = "b")` and `set_prior("normal(0,10)", class = "b", coef = "x1")` at the same time. This will set a $\text{normal}(0,10)$ prior on the effect

of $x1$ and a $\text{normal}(0,2)$ prior on all other population-level effects. However, this will break vectorization.

The intercept has its own parameter class named “Intercept” and priors can thus be specified via `set_prior("", class = "Intercept")`. Setting a prior on the intercept will not break vectorization of the other population-level effects. Note that technically, this prior is set on an intercept that results when internally centering all population-level predictors around zero to improve sampling efficiency. On this centered intercept, specifying a prior is actually much easier and intuitive than on the original intercept, since the former represents the expected response value when all predictors are at their means. To treat the intercept as an ordinary population-level effect and avoid the centering parameterization, use $0 + \text{intercept}$ on the right-hand side of the model formula.

A special shrinkage prior to be applied on population-level effects is the horseshoe prior. See `horseshoe` for details. Another shrinkage prior is the so-called lasso prior. See `lasso` for details.

In non-linear models, population-level effects are defined separately for each non-linear parameter. Accordingly, it is necessary to specify the non-linear parameter in `set_prior` so that priors can be assigned correctly. If, for instance, `alpha` is the parameter and `x` the predictor for which we want to define the prior, we can write `set_prior("", coef = "x", nlnpar = "alpha")`. As a shortcut we can use `set_prior("", nlnpar = "alpha")` to set the same prior on all population-level effects of `alpha` at once.

If desired, population-level effects can be restricted to fall only within a certain interval using the `lb` and `ub` arguments of `set_prior`. This is often required when defining priors that are not defined everywhere on the real line, such as uniform or gamma priors. When defining a $\text{uniform}(2,4)$ prior, you should write `set_prior("uniform(2,4)", lb = 2, ub = 4)`. When using a prior that is defined on the positive reals only (such as a gamma prior) set `lb = 0`. In most situations, it is not useful to restrict population-level parameters through bounded priors (non-linear models are an important exception), but if you really want to this is the way to go.

23.3.2 Standard deviations of group-level ('random') effects

Each group-level effect of each grouping factor has a standard deviation named `sd_`. Consider, for instance, the formula $y \sim x1 + x2 + (1 + x1 | g)$. We see that the intercept as well as $x1$ are group-level effects nested in the grouping factor `g`. The corresponding standard deviation parameters are named as `sd_g_Intercept` and `sd_g_x1` respectively. These parameters are restricted to be non-negative and, by default, have a half student-t prior with 3 degrees of freedom and a scale parameter that depends on the standard deviation of the response after applying the link function. Minimally, the scale parameter is 10. This prior

is used (a) to be only very weakly informative in order to influence results as few as possible, while (b) providing at least some regularization to considerably improve convergence and sampling efficiency. To define a prior distribution only for standard deviations of a specific grouping factor, use `set_prior("", class = "sd", group = "")`. To define a prior distribution only for a specific standard deviation of a specific grouping factor, you may write `set_prior("", class = "sd", group = "", coef = "")`. When defining priors on group-level parameters in non-linear models, please make sure to specify the corresponding non-linear parameter through the `nlpars` argument in the same way as for population-level effects.

23.3.3 Correlations of group-level ('random') effects

If there is more than one group-level effect per grouping factor, the correlations between those effects have to be estimated. The prior “`lkj_corr_cholesky(eta)`” or in short “`lkj(eta)`” with $\eta > 0$ is essentially the only prior for (Cholesky factors) of correlation matrices. If $\eta = 1$ (the default) all correlations matrices are equally likely a priori. If $\eta > 1$, extreme correlations become less likely, whereas $0 < \eta < 1$ results in higher probabilities for extreme correlations. Correlation matrix parameters in brms models are named as `cor_`, (e.g., `cor_g` if `g` is the grouping factor). To set the same prior on every correlation matrix, use for instance `set_prior("lkj(2)", class = "cor")`.

23.3.4 Splines

Each spline has its corresponding standard deviations modeling the variability within this term. This parameter class is called `sds` and priors can be specified via `set_prior("", class = "sds", coef = "")`. The default prior is the same as for standard deviations of group-level effects.

23.3.5 Gaussian processes

Gaussian processes have two parameters, the standard deviation parameter `sdgp`, and characteristic length-scale parameter `lscale` (see `gp` for more details). The default prior of `sdgp` is the same as for `sd`-s of group-level effects. The default prior of `lscale` is an informative inverse-gamma prior specifically tuned to the covariates of the Gaussian process (for more details see https://betanalpha.github.io/assets/case_studies/gp_part3/part3.html). This tuned prior may be overly informative in some cases, so please consider other priors as well to make sure inference is robust to the prior specification. If tuning fails, a half-normal prior is used instead.

23.3.6 Autocorrelation parameters

The autocorrelation parameters are named ar (autoregression), ma (moving average), arr (autoregression of the response), car (spatial conditional autoregression), as well as lagsar and errorsar (Spatial simultaneous autoregression). Priors can be defined by `set_prior("", class = "ar")` for ar and similar for other autocorrelation parameters. By default, ar and ma are bounded between -1 and 1, car, lagsar, and errorsar are bounded between 0, and 1, and arr is unbounded (you may change this by using the arguments `lb` and `ub`). The default prior is flat over the definition area.

23.3.7 Parameters for specific families

Families `gaussian`, `student`, `skew_normal`, `lognormal`, and `gen_extreme_value` need the parameter `sigma` to account for the residual standard deviation. By default, `sigma` has a half student-t prior that scales in the same way as the group-level standard deviations. Family `student` needs the parameter `nu` representing the degrees of freedom of students-t distribution. By default, `nu` has prior `"gamma(2, 0.1)"` and a fixed lower bound of 1.

Families `gamma`, `weibull`, `inverse.gaussian`, and `negbinomial` need a shape parameter that has a `"gamma(0.01, 0.01)"` prior by default.

Every family specific parameter has its own prior class, so that `set_prior("", class = "")` is the right way to go. All of these priors are chosen to be weakly informative, having only minimal influence on the estimations, while improving convergence and sampling efficiency. Often, it may not be immediately clear, which parameters are present in the model. To get a full list of parameters and parameter classes for which priors can be specified use function `get_prior`.

```
library(tidyverse)
library(mice)
```


Chapter 24

Puuduvad andmed

Oletame, et meil on andmetabel, mis sisadab lisaks sisestatud andmetele ka NA-sid ja mille pealt te tahate fittida regresionimudeli. NA-dega tabeleid tuleb muidugi ette ka mujal, aga just mitmesel regressioonil on nad eriline probleem, sest mudeli fittimisel saab kasutatada ainult neid tabeli ridu, mis ei sisalda ühtegi NA-d. Seega on meie huvides asendada NA-d mingite enam-vähem mõistlikke väärustega, sest vastasel juhul võib meie efektiivne valim väga kokku kuivada, mis omakorda viib alla analüüsni sensitiivsuse. Sellist NA-de asendamist nimetame imputatsiooniks ja algoritmi, mille abil me NA-d väärustega asendame, kutsume imputatsionimudeliks. Imputatsiooni tulemuseks on üks või mitu andmetabelit, mis sisaldavad nii empiirilisi kui imputeeritud väärusti ja reeglini ei sialda enam NA-sid. Nende tabelite pealt fitime siis oma päris regresionimudeli. Imputatsionimudel on sageli samuti regresionimudel, aga see ei pruugi olla sama mudel samade muutujatega, kui hilisem "päris" regresionimudel. Imputeeritud tabelilt ootame, et selle pealt arvutatud keskmised, varieeruvused jm mudeli koefitsiendid oleksid võimalikult sarnased müütilise tabeliga, kus ei olnud juba alguses puuduvaid väärusti (mitte tegeliku NA-dega algtabeliga).

Alustuseks mõned kasulikud nipid:

- Järgnev koodirida tagab, et lm() funktsioon teeb automaatselt drop_na():
`options(na.action = na.omit)`
- `na.action(fit)` annab lm() mudeli objekti pealt tabelist välja visatud ridade numbrid
- `naprint(na.action(fit))` annab selliste ridade arvu
- `colSums(is.na(airquality))` annab veeru kaupa NA-de arvu
- `md.pattern(data, plot = FALSE)` annab NA-de paiknemise tabeli rea ja veeru kaupa.

- VIM::aggr(data, prop = FALSE, numbers = TRUE) ja VIM::matrixplot(data) joonistavad NA-de mustrist pildi.
- dfs[is.na(dfs)] <- 0 rekodeerib NA-d nullideks
- na_if(dfs, 0) teeb 0-d NA-deks
- drop_na(data, c(column1, column2)) viskab välja NA-sid sisaldavad read 2st veerust.
- filter_all(weather, any_vars(is.na(.))) näitab ridu, mis sisal-davad NA-sid

Nii saab 2 vektori põhjal kolmanda nii, et NA-d asendatakse vastava väärtsusega:

```
y <- c(1, 2, NA, NA, 5)
z <- c(NA, NA, 3, 4, 5)
coalesce(y, z)
#> [1] 1 2 3 4 5
```

viska tabelist välja veerud, milles on >80% NA-sid

```
vekt <- sapply(df, function(x) mean(is.na(x)))
#NA-de protsent igas veerus
vekt
vekt1 <- nad[nad < 0.8]
#subsettisin vektori elemendid, mis on < 0.8 (NA-sid alla 80%). 216 tk.
#nad1 is a named vector
vekt1n <- names(vekt1) #vektor named vektori vekt1 nimedest
df_with_fewer_cols <- subset(df, select = vekt1n)
#subsetime (jätame alles) ainult need df-i veerud,
#mille nimele vastab mõni vektori nad1n element
```

24.1 Andmete puudumise mehhanismid:

1. *missing completely at random (MCAR)* - NA-d on jaotunud juhuslikult – põhimõtteliselt ei ole võimalik ennustada, et tabeli mingis positsioonis on NA erineva tõenäosusega kui igas teises. Ainult MCAR andmete korral on õigustatud NA-dega riidade eemaldamine. Igal pool mujal kallutab see pahasti teie poolt arvutatud statistikute väärtsusi.
2. *missing at random (MAR)* - Andmed on jagatavad gruppidesse ja iga gruup sees on NA-d juhuvalimina. Kui meil on NA-d muutujas A, siis andmete puudumine ei sõltu A väärtestest, aga ta võib sõltuda muutujate B, C, jne väärtestest. Kaasaegsed imputatsioonimeetodid kasutavad enamasti MAR-eeldust, mis omakorda eeldab, et relevantsed muutujad on hõlmatus imputatsioonimudelisse (aga mitte tingimata regressioonimudelisse, mis fititakse imputeeritud andmetel).

3. *missing not at random (MNAR)* - NA-d muutujas A sõltuvad ainult A väärustusest. Kui andmed ei ole MCAR ega MAR, siis me eeldame, et nad on MNAR. MNAR imputatsioon vajab eraldi mudelit, mis arvestab NA-sid genereeriva mehhanismiga — see on raske asi.

Seda, millisesse kategooriasse mingi andmeset kuulub, ei saa üldjuhul öelda andmete endi põhjal. Selleks tuleb tunda andmeid genereerivat mehhanismi. Seega on oma andmete paigutamine ühte kolmest kategooriast praktikas sageli üsna ebakindel.

24.2 Ühekaupa imputatsiooni meetodid

Siin käitleme meetodeid, mis annavad tulemuseks ühe imputeeritud tabeli, kus iga NA asemel on üks ja ainult üks number. Neid meetodeid on lihtne mõista, aga nende kasutamist tuleks enamasti vältida.

24.2.1 NA-dega ridade eemaldamine

Pealiskaudsel vaatlusel võib tunduda, et see on üks hea ja mittekallutatud meetod, eriti siis kui meil on andmetes vaid väike NA-de osakaal. Tegelikult see ei ole nii. NA-dega ridade eemaldamist (drop_na) on mõistlik kasutada vaid kolmel erijuuhul:

1. MCAR juhul ei kalluta see keskmist, regressioonikoeffitsiente ega korrelatsiooni, kuid vähendab analüüsni sensitiivsust. Kui laseme regressiooni paljude X-muutujatega, millega on kasvõi väike osa NA-sid, siis nende juhuslikud kombinatsioonid viivad paraku enamuse ridade eemaldamisele tabelist ja massiivsele informatsioonikaole.
2. Kui meil on NA-d ainult Y-muutujas (ehk muutujas, mida soovime regressioonimudelis ennustada), siis on NA-dega ridade eemaldamine sama hea kui mitmene imputatsioon ka MAR juhul (eeldades, et mitmene imputatsioon kasutab samu x-muutujaid, mis imputeeritud andmetega regressioon).
3. logistilise mudeli korral on NA-dega ridade väljaviskamine eelistatud meetod juhul kui NA-d esinevad ainult dihhotoomses Y või dihhotoomses X muutujas (aga mitte mõlemas) ja tõenäosus kohata NA-d sõltub ainult Y-väärtusest (mitte X-st).

24.2.2 Paariviisiline deleteerimine

Kui muutujad on multivariaatselt normaalsed, nende vahel ei ole korrelatsioone ja kehtib MCAR eeldus, siis on hea kasutada *pairwise deletion e available case*

meetodit, mis arvutab igale muutujale olemasolevate andmete pealt keskmise ja SEM-i ja kasutab korrelatsioonide jms arvutamisel kõiki ridu, kus andmepaarid on täielikud. Siinkohal tuleb küll mainida, et sellist olukorda praktikas naljalt ette ei tule.

Regressioonil tuleb meil lm() asemel kasutada lavaan raamatukogu.

```
data <- airquality[, c("Ozone", "Solar.R", "Wind")]
mu <- colMeans(data, na.rm = TRUE)
cv <- cov(data, use = "pairwise")

library(lavaan)
fit <- lavaan("Ozone ~ 1 + Wind + Solar.R
              Ozone ~~ Ozone",
              sample.mean = mu, sample.cov = cv,
              sample.nobs = sum(complete.cases(data)))
```

24.2.3 Keskmise imputatsioon ja regressiooniga imputatsioon

Kasutame mice raamatukogu.

```
imp <- mice(airquality, method = "mean", m = 1, maxit = 1)
```

See on kindel viis varieeruvust alahinnata ja tekitada kummalisi andmejaotusi. Kui meil on MCAR andmed, siis tuleb vähemalt keskmise hinnang nikketa, aga pea kõik teised statistikud lähevad puusse. Seda meetodit tuleks alati vältida.

Regressiooniga imputatsioonil fitime mudeli olemasolevatel andmetel ja seejärel ennustame seda kasutades piuduvad andmepunktid

```
fit <- lm(Ozone ~ Solar.R, data = airquality)
pred <- predict(fit, newdata = ic(airquality))
```

See on jällegi hea viis, kuidas kunstlikult andmete varieervust vähendada ja nende jaotust muuta. Ühtlasi töstab see meetod kunstlikult muutujate vahelist korrelatsiooni. Enam halvemaks ei saa minna!

24.2.4 Stohastiline regressiooniga imputatsioon

lisab eelnevasse müra ja töötab ka MAR andmete peal. Kõigepealt fitib lm()-ga mudeli koefitsiendid, siis ennustab nende pealt piuduvad väärtsused ja lõpuks liidab igale neist juhusliku residuaali algsest mudelist.

```
data <- airquality[, c("Ozone", "Solar.R")]
imp <- mice(data, method = "norm.nob", m = 1, maxit = 1,
            seed = 1, print = FALSE)
```

Eeldus, mis kunagi ei kehti, on et mudeli koefitsiendid vastavad tegelikkusele. Seega puudub siin koefitsientide määramise ebakindlus (seda saab arvutada kas Bayesi meetoditega tõmmates parameetrväärtuste ebakindluse otse posteerioritest, või bootstrappides).

Stohastiline regressiooniga imputatsioon võib anda negatiivseid väärtusi muidu positiivsetesse muutujatesse, ekstreemsed väärtused pole hästi mudeliga kaetud ja ennustused/imputatsioonid vastavad regressioonieeldustele (homoskedastilisus jne) ka siis, kui andmed nendele ei vasta. Siiski, kuna see pigem algeline meetod säilitab korrelatsioonid muutujate vahel ja ei kalluta regressioonikoefitsiente, ei saa see liiga halb olla. Sellelt põhjalt on ehitatud paljud kaasaegsed imputatsionimeetodid.

24.2.5 Eelmise vaatluse kopeerimine (LOCF).

See on *ad hoc* meetod aegridadele, mida rakendab `tdy::fill()` ja mida on hea kasutada siis, kui muutuja väärtus kirjutatakse üles ainult siis, kui see muutub. Muidu, LOCF võib olla kallutatud isegi MCAR juhul.

24.3 Mitmene imputatsioon

See meetod loodi Donald Rubini poolt 1970ndatel üldskeemiga:

1. Tekita mitu imputeeritud andmeraami (igaüks on ilma NA-deta), mis erinevad ainult imputeeritud väärtuste poolest. Iga NA asemele imputeeritakse arvud omast jaotusest, mille laius arvestab meie ebakindlusega selle konkreetse positsiooni imputeerimisel.
2. Analüüs ihat andmeraami eraldi, aga tavapärasel viisil
3. kombineeri tulemused lõplikuks keskmiseks ja sem-iks.

Näide: mitmene imputatsioon `lm()`-ga ja CI-d koefitsientidele

```
imp <- mice(nhanes, print = FALSE, m = 10, seed = 24415)
fit <- with(imp, lm(bmi ~ age))
est <- pool(fit)
summary(est, conf.int = TRUE)

complete(imp, 3) #annab 3. imputeeritud tabeli (nii uued kui orig andmed)
complete(imp, "long") #annab kõik imputeeritud tabelid pikas formaadis
complete(imp, "broad") #sama laias formaadis
imp$data #originaalsed andmed NAdega
imp$imp #ainult imputeeritud andmed
```

`m=10` loob 10 iseseisvat andmeraami.

Kaasaegne soovitus on, kui võimalik, sättida m kuhugi 20-100 vaheline.

Kogu järgnev imputatsioonijutt käib vaikimisi mitmese imputatsiooni kohta.

Imputatsiooni mudel peaks

- arvestama protsessiga, mis genereeris NAd (NMAR juhul)
- säilitama andmetes leiduvad omavahelised suhted
- säilitama nende suhete ümber laiuva ebakindluse määra

Selle nimel peame tegema 7 valikut selles järjekorras:

1. **Kas me loeme MAR eelduse kehtivaks?** MNAR imputatsioon toob sisse lisaeeldused. MNAR juhul on kaks võimalust: toome sisse uued muutujad, mis enustaks NAd paiknemist lootuses, et läheneme MAR-eelduse täitmisele; või kasutame lisamudelit, mis arvestab protsessiga, mis genereeris NAd.
2. **Milline imputatsionimudel (nii mudeli struktuurne osa kui eel-datud vigade jaotus)?** *Fully conditional specification* (FCS) imputeerib multivariatseid NA-sid muutuja haaval. Seega peab mudeli kuju spetsifitseerima igale NA-dega muutujale eraldi, arvestades ka muutujate vahelisi seoseid.

Method	Description	Scale Type
pmm	Predictive mean matching	Any*
midastouch	Weighted predictive mean matching	Any
sample	Random sample from observed values	Any
cart	Classification and regression trees	Any
rf	Random forest imputation	Any
norm	Bayesian linear regression	numeric
norm.boot	Normal imputation with bootstrap	Numeric
quadratic	Imputation of quadratic terms	Numeric
ri	Random indicator for nonignorable data	Numeric
logreg	Logistic regression	Binary*
logreg.boot	Logistic regression with bootstrap	Binary
polr	Proportional odds model	ordinal*
polyreg	Polytomous logistic regression	Nominal*

* - vaikemudel antud andmetüübile

Nii saab trükkida kasutatud imputatsioonimeetodi veeru kaupa:

```
imp$meth
methods(mice) #annab mice meetodite loendi

ini <- mice(nhanes2, maxit = 0)
meth <- ini$meth
```

```
meth["bmi"] <- "norm" #muudab meetodit muutujale "bmi"
imp <- mice(nhanes2, meth = meth, print=F)
```

Normaalsetele muutujatele on norm efektiivsem, kui pmm. norm.boot on kiirem mitte-bayesiaanlik alternatiiv. Kui pmm töötab halvasti (näit ei ole piisavalt ennustuse lähedasi doonorarve), kasuta norm meetodeid. Harvade kategooriliste andmepunktide korral eelista pmm-i, pigem kui logreg, polr või polyreg. sample on kiirmeetod imputeerimiseks ilma kovariaatideta.

3. **Milliseid muutujaid kasutada imputatsionimudelis prediktoritenä?** Üldine soovitus on lisada kõik relevantsed muutujad ja nende relevantsed interaktsioonid.

`predictorMatrix` argument on maatriks suurusega `ncol(data)`, mis sisaldab 0/1. Iga rida ütleb, milliseid prediktoreid kasutatkse muutujale, mille nimi on reanimes.

```
imp <- mice(nhanes, print = FALSE)
imp$predictorMatrix
#>      age bmi hyp chl
#> age    0   1   1   1
#> bmi    1   0   1   1
#> hyp    1   1   0   1
#> chl    1   1   1   0
```

Kõikide muutujate kasutamine on ok, kui meil on $< 20\text{-}30$ muutuja, mille vahel ei ole keerulisi interaktsioone. Rohkem muutujaid muudab ka MAR eelduse täitmise tõenäolisemaks. Muidu võta sisse (i) kogu hiljem kasutatava regressioonimudeli, (2) lisa muutujad, mis võiksid ennustada NA-de teket (mille jaotused erinevad andmete ja NA-de korral imputeeritavas muutujas), (3) lisa muutujad, mis on korreleeritud imputeeritava muutujaga. `quickpred()` annab kiire viisi prediktormaatriksi defineerimiseks.

Nii saab prediktor maatriksi ilma tegelikku impoutatsiooni läbi tegemata:

```
ini <- mice(nhanes, maxit=0, print=F)
pred <- ini$pred
pred[ , "hyp"] <- 0 #rekodeerime veeru "hyp",
#mida ei kasutata enam imputatsioonil
imp <- mice(nhanes, pred=pred, print=F)
```

`quickpred()` teeb ise prediktorite valiku, mida imputatsioonil kasutada

```
ini <- mice(nhanes, pred=quickpred(nhanes, mincor=.3), print=F)
ini$pred
```

Nii kirjutame imputatsionimudelisse sisse interaktsiooni kahele muutujale, kus me kõigepealt lahutame kummagist muutujast tema keskmise (tsentreerime) ja siis korrutame need muutujad

```

expr <- expression((wgt - mean(wgt)) * (hc - mean(hc)))
boys$wgt.hc <- with(boys, eval(expr))
meth <- make.method(boys)
meth[["wgt.hc"]] <- paste("~I(", expr, ")", sep = "")
#I() lubab argumendi (wgt - mean(wgt)) * (hc - mean(hc))
#ära tunda literaase valemina.
meth[["bmi"]] <- ""
pred <- make.predictorMatrix(boys)
pred[c("wgt", "hc"), "wgt.hc"] <- 0
#et katkestada võimalikke tagasisidestusega luupe wgt.hc ja
#wgt&hc vahel, viime sisse muutused prediktormaatriksisse.
imp.int <- mice(boys, m = 1, meth = meth, pred = pred,
                 print = FALSE, seed = 62587, maxit = 10)

```

Kategooriliste muutujate interaktsioone saab imputeerida jagades andmed gruppidesse ja imputeerides neid eraldi, misjärel kombineerida tulemused taas rbind() abil.

Kompositsoonilised andmed – mitu muutujat summeeruvad ühele (100%-le). Selliste andmete imputeerimiseks vt robCompositions raamatukogu.

4. Kas imputeerida muutujaid, mis on teiste NA-dega muutujate funktsioonid (suhted jms)? Lihtsaim vastus on “ei”. Siin imputeerime kõigepealt wght ja hgth muutujad ja siis arvutame nende pealt suhte (whr).

```

data <- boys[, c("age", "hgt", "wgt", "hc", "reg")]
imp <- mice(data, print = FALSE, seed = 71712)
long <- mice::complete(imp, "long", include = TRUE)
long$whr <- with(long, 100 * wgt / hgt)
imp.itt <- as.mids(long)

```

See ei ole täiuslik meetod: päris regressioonimudelis saavad kõikide muutujate koefitsiendid, mis sõltuvad whr-ist, kallutatud nulli suunas.

Alternatiivne meetod on nn **passiivne imputatsioon**, kus transformatsioon tehakse koos imputatsiooniga samas algoritmis. Passiivseks imputatsiooniks on vaja alustada imputatsioonimeetodi defineerimist ~ (tilde) sümboliga.

```

data <- boys[, c("age", "hgt", "wgt", "hc", "reg")]
data$whr <- 100 * data$wgt / data$hgt
meth <- make.method(data)
meth[["whr"]] <- "~I(100 * wgt / hgt)"
#I() lubab argumendi 100*wght/hgth ära tunda literaase valemina.
pred <- make.predictorMatrix(data)
pred[c("wgt", "hgt"), "whr"] <- 0
#et katkestada võimalikke tagasisidestusega luupe whr-i ja
#wgt&hgt vahel, viime sisse muutused prediktormaatriksisse.
imp.pas <- mice(data, meth = meth, pred = pred, print = FALSE)

```

whr imputatsioonid saadakse hgt ja wgt-st vastavalt $100 * \text{wght} / \text{hgth}$ valemile. Kuna whr on andmetabeli viimane veerg, imputeeritakse see peale wgt ja hgt-d. Kahjuks kallutab ka passivne imputasioon koefitsiente nulli suunas. Suhteid ongi raske imputeerida!

Suhete, ruutliikmete ja interaktsioonide imputeerimiseks on arvatavasti parim meetod järgmine:

```
library(smcfcs)
data <- pop
data[sample(nrow(data), size = 100), "wgt"] <- NA
data[sample(nrow(data), size = 100), "hgt"] <- NA
data$whr <- 100 * data$wgt / data$hgt
meth <- c("", "norm", "norm", "", "", "norm")
imps <- smcfcs(originaldata = data, meth = meth, smtype = "lm",
                 smformula = "hc ~ age + hgt + wgt + whr")
fit <- lapply(imps$impDatasets, lm,
               formula = hc ~ age + hgt + wgt + whr)
summary(pool(fit))
```

5. Mis järjekorras muutujaid imputeerida?

Tavaliselt imputeeritakse muutujaid järjekorras vasakult paremale ja see on hea küll. Imputeerimise järjekorrale tasub tähelepanu põõrata siis, kui NA-de muster on monotooniline. NA-de muster on monotooniline siis, kui muutujaid saab järjestada nõnda, et kui esimeses muutujas on NA, siis ka kõikides järgnevates muutujates on selles reas NA. (Näiteks, kui aegreas mõni subjekt välja kukub.)

Näiteks siin on meil 3 monotoonilist muutujat

```
data <- nhanes2[, 1:3]
md.pattern(data, plot = FALSE)
#>   age hyp bmi
#> 16   1   1   1   0
#> 1    1   1   0   1
#> 8    1   0   0   2
#>     0   8   9  17
```

Meil on 16 ilma NA-deta rida, 1 rida, kus puudub bmi üksi ja 8 rida, kus puuduvad bmi ja hyp. Paremas servas on veerg rea kaupa NA-de arvuga igale kombinatsioonile ja alumine rida annab NA-de arvu igale veerule ja totaalse NAd arvu (17).

```
imp <- mice(data, visit = "monotone", maxit = 1, m = 2,
             print = FALSE)
```

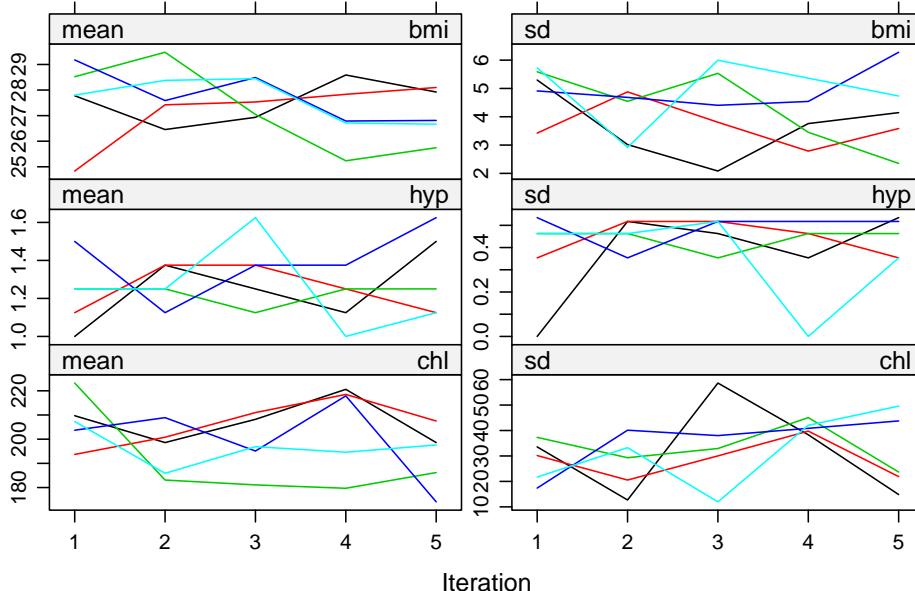
Argument `visit = monotone` määrab monotoonilise imputatsiooni, ehk kõigepealt imputeeritakse need muutujad, milles on vähem NA-sid (enne hyp ja siis bmi). Monotooniline imputatsioon eeldab, et NA-de muster on monotooniline ja

et muutujad ei sõltu üksteisest (muutujate distinktsust).

6. Mitu iteratsiooni teha?

Nii saab mõõtmeid agelate konvergeerumist graafiliselt kontrollida.

```
plot(imp)
```



```
mice.mids(imp, maxit=35, print=F)
#> Class: mids
#> Number of multiple imputations: 5
#> Imputation methods:
#>   age   bmi   hyp   chl
#>   "" "pmm" "pmm" "pmm"
#> PredictorMatrix:
#>   age   bmi   hyp   chl
#>   age   0     1     1     1
#>   bmi   1     0     1     1
#>   hyp   1     1     0     1
#>   chl   1     1     1     0
#tõstab mõõtmeid iteratsioonide arvu 35-le.
#võtab imputeeritud mids objekti ja lisab iteratsioone,
#toodab uue mids objekti
```

Joonisel on igale iteratsioonile imputeeritud väärustete (aga mitte algsete väärustete) keskmise ja SD.

7. Mitu imputatsiooni teha? Liiga suur m viib suurele simulatsiooniveale ja statistilisele ebaefektiivsusele. Vaikeväärustus on 5.

24.4 Predictive mean matching

See on enimkasutatud mitmese imputatsiooni meetod:

1. ennusta NA asemele parim oodatud väärthus regressioonimudelist
2. leia mingi arv (3, 5 v 10) oodatud väärtsusega sarnaseid kandidaatväärtsusi (ehk doonorväärtsusi $d=$) olemasolevatest andmetest
3. võta nende seast üks juhuslik väärthus ja imputeeri see.

See põhineb eeldusel, et kandidaatväärtsused on sama jaotusega, kui puuduvad väärtsused. Meetod on robustne andmete tranformatsioonidele ja töötab üldiselt hästi pidevatel muutujatel, juhul kui valim on suur. Kui $N > 100$, kasuta $d=5$, väikeste valimite korral aga $d=1$. Kui N on ligikaudu 10, siis on eelis adaptatiivsel meetodil **midastouch**.

Predictive mean matching $d=5$ on mice vaikemudel pidevatele muutujatele. Vaikemudel on lineaarne additiivne mudel üle kõikide prediktorite. Kui päris regressioonimudel on vaikemudelist erinev (interaktsioonid!), siis tuleks seda kasutada ka imputatsioonil. Meetod võib töötada halvasti väikesel valimitel, seda ei saa kasutada andmetest välja ekstrapoleerimiseks ja ka andmete sees, juhul kui andmete tihedus on madal.

24.5 Imputeerimine mitte-normaalsete jaotuste korral

Kui $n > 400$, pole enamasti probleemi imputeeritud andmete pealt keskmise arvutamisega (isegi kui muutujas on 75% NA-sid), kuid varieeruvusnäitajad võivad siiski olla problemaatilised. Üks võimalus on transformeerida andmed nii, et nad oleksid normaalsed, siis imputeerida ja seejärel andmed uesti tagasi transformeerida. Siiski ei ole garantii, et selline trikitamine asja hoopis hullemaks ei tee. Teine võimalus on imputeerida, kastudades mitte-normaalseid jaotusi. Näiteks saab kasutada studenti t jaotust, et saada imputatsioonil robustsemaid tulemusi, juhul kui andmetes on outlierid, mis kajastavad tegelikke olusid.

```
library("ImputeRobust")
library("gamlss")
imp <- mice(data2, m = 1, meth = "gamlssTF", print = FALSE)
```

24.6 Kategoorilised muutujad

Binaarse muutuja korral kasutame logistilist mudelit, rohkem kui kahe tase-mega mittejärjestatud juhul multinominaalset logistilist mudelit ja järjestatud

tasemetega juhul *ordered logit* mudelit või *proportional odds* mudelit.

Kategooriliste muutujate imputeerimine on raske. Logistilises regressioonis vajame me reeglina vähemalt 10 sündmust prediktori kohta. Seega, kui me imputeerime 10 binaarset NAd, vajame me 100 sündmust ja kui sündmuse tõenäosus on 0.1, siis üle 1000 katse. Kui muutuja tasemete arv kasvab, siis need numbrid kasvavad väga kiiresti.

Vaikimisi töötavad logreg, polyreg ja polr on tavaliselt ok. Kui suhe sündmuste arv/fititud parameetrite arv langeb alla 10, siis võiks kasutada robustsemaid meetodeid: pmm, cart või rf.

24.7 countid (sisaldavad nulle)

Meetodid:

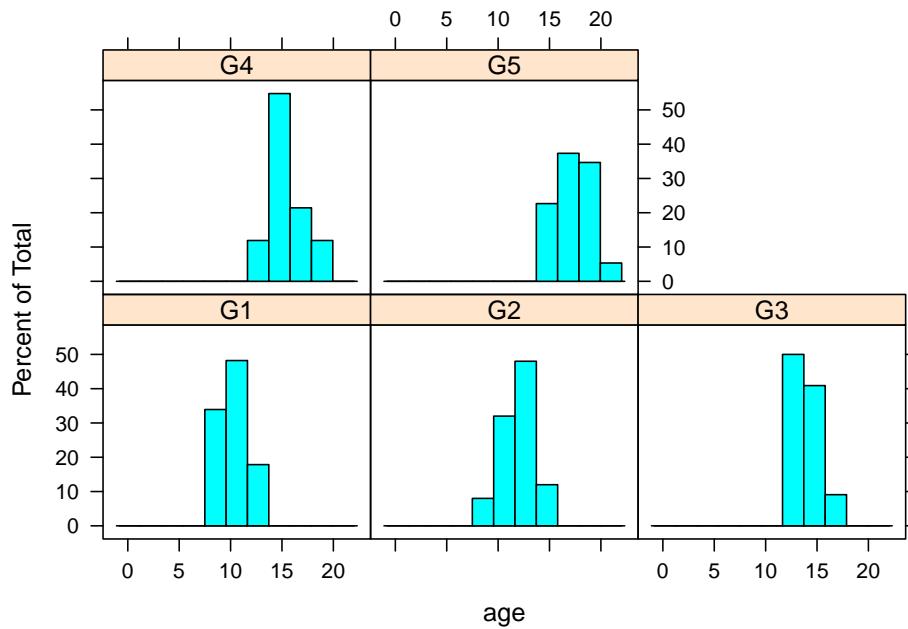
1. Predictive mean matching.
2. Ordered categorical imputation.
3. (Zero-inflated) Poisson regression.
4. (Zero-inflated) negative binomial regression.

Poisson mudeldab haruldaste sündmuste summat, negatiivne binommuodel on paindlikum versioon Poissonist, mida saab kasutada siis, kui meil on andmetes üledispersioon (poisson eeldab kindlat dispersioonimäära). Zero-inflated versioonid mölemast töötavad siis, kui meil on andmetes oodatust rohkem nulle.

ImputeRobust raamatukogu kasutab mice meetodeid: `gamlssPO` (Poisson), `gamlssZIBI` (zero-inflated binomial) ja `gamlssZIP` (zero-inflated Poisson).

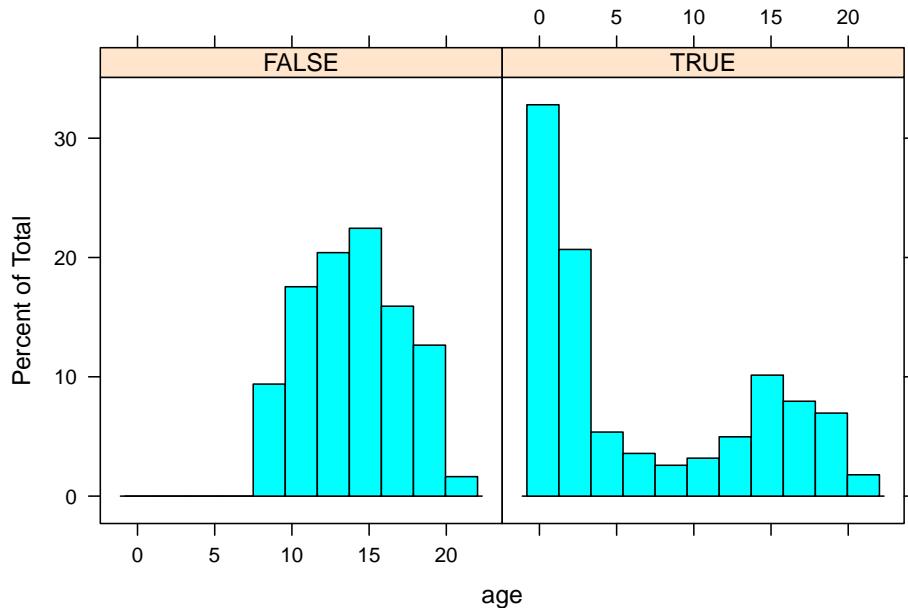
24.8 Graafilised meetodid

konditsionaalne plot muutujale “age” erinevatel muutuja “gen” väärustel
`histogram(~age|gen, data=boys)`



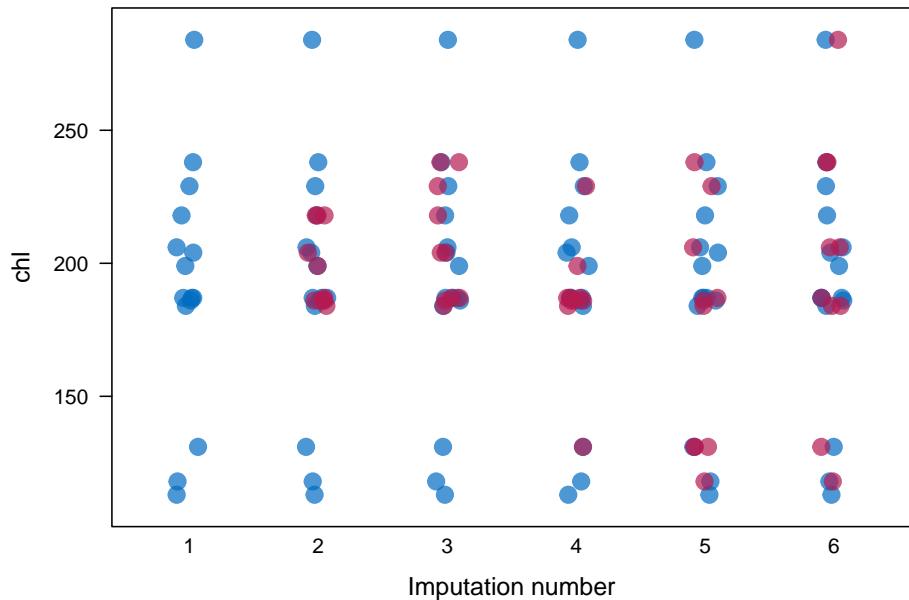
Vanuste jaotus on erinev sõltuvalt sellest, kas gen muutuja on NA v mitte.

```
R <- is.na(boys$gen)
histogram(~age|R, data=boys)
```



näitab algseid ja imputeeritud andmeid üksteise peal muutujale "chl"

```
stripplot(imp, chl~.imp, pch=20, cex=2)
```



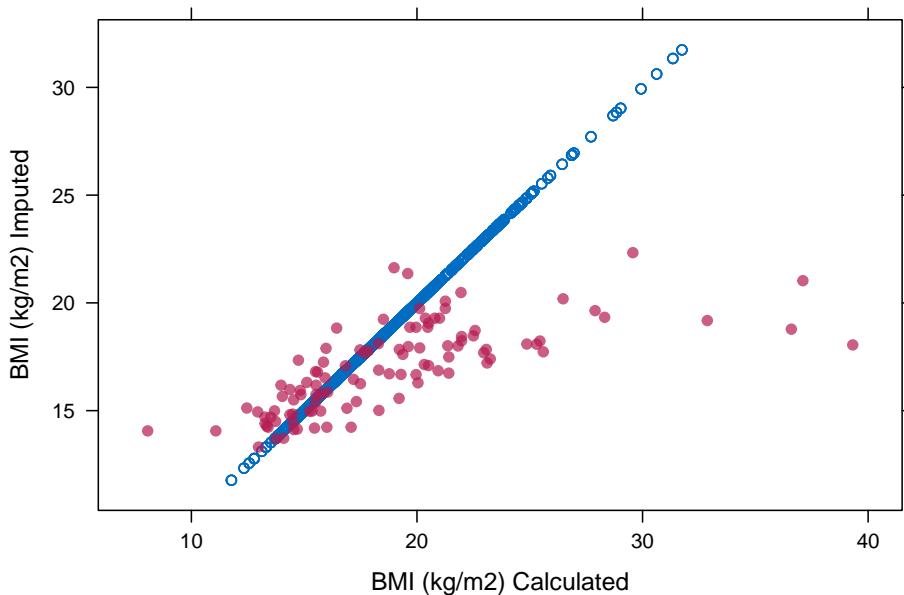
Imputeeritud andmed on punased.

suru imputeeritud andmed 1 ja 25 vahel

```
ini <- mice(boys, maxit = 0)
meth <- ini$meth
meth["tv"] <- "norm"
post <- ini$post
post["tv"] <- "imp[[j]][, i] <- squeeze(imp[[j]][, i], c(1, 25))"
imp <- mice(boys, meth=meth, post=post, print=FALSE)
```

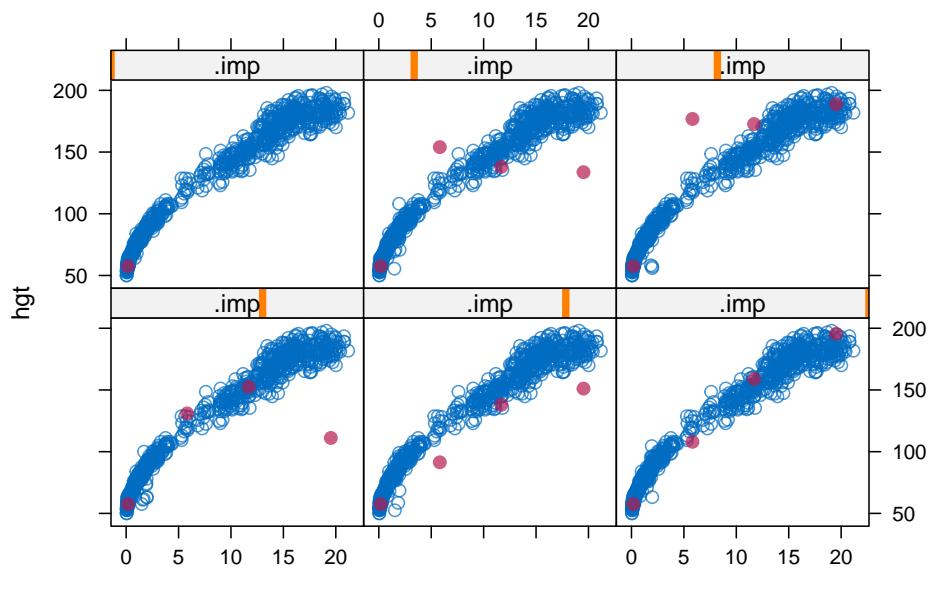
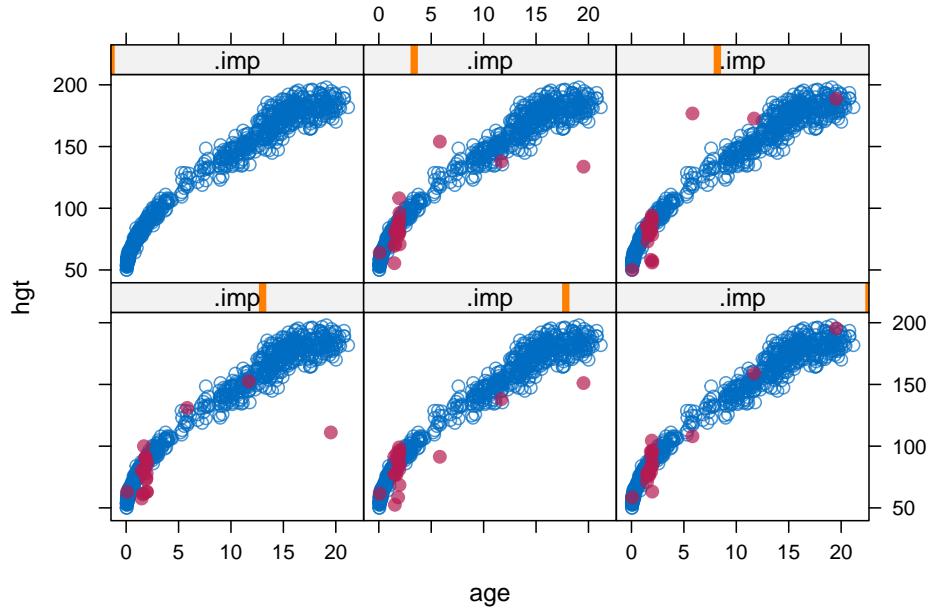
scatterplot, kus on nii imputeeritud kui päris andmed

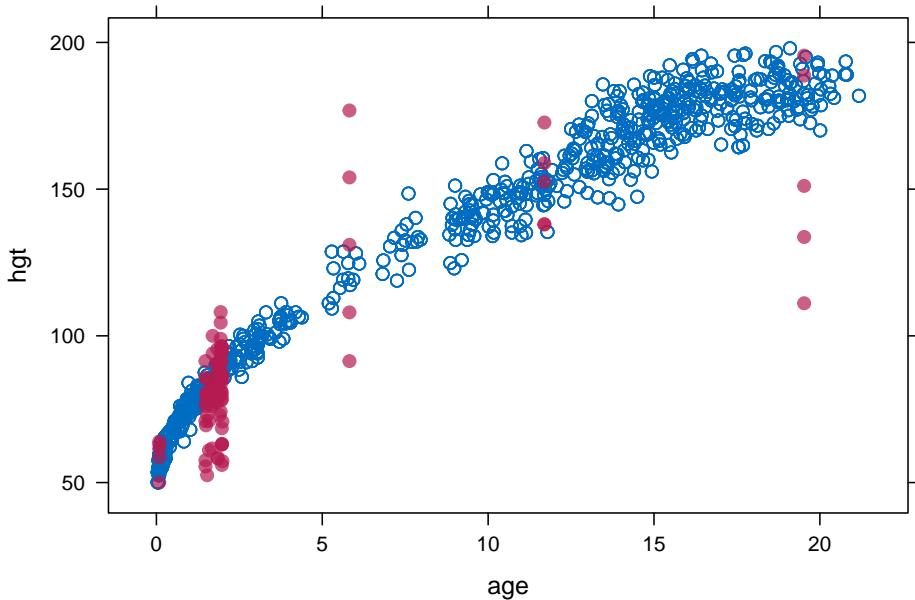
```
miss <- is.na(imp$data$bmi)
xyplot(imp, bmi ~ I (wgt / (hgt / 100)^2),
na.groups = miss, cex = c(0.8, 1.2), pch = c(1, 20),
ylab = "BMI (kg/m2) Imputed", xlab = "BMI (kg/m2) Calculated")
```



```
imp <- mice(boys, maxit=1)
#>
#>   iter imp variable
#>   1   1   hgt   wgt   bmi   hc   gen   phb   tv   reg
#>   1   2   hgt   wgt   bmi   hc   gen   phb   tv   reg
#>   1   3   hgt   wgt   bmi   hc   gen   phb   tv   reg
#>   1   4   hgt   wgt   bmi   hc   gen   phb   tv   reg
#>   1   5   hgt   wgt   bmi   hc   gen   phb   tv   reg

### xyplot: scatterplot by imputation number
### observe the erroneous outlying imputed values
### (caused by imputing hgt from bmi)
xyplot(imp, hgt~age|imp, pch=c(1,20),cex=c(1,1.5))
```





24.9 Tulemuste avaldamine

Valdatud töö võiks sisalda vastuseid järgmistele küsimustele

1. Kui palju NA-sid on igas muutujas? Mis oleks valimi suurus *complete-cases* analüüsile, kui me teeme `drop_na()`? Kui NA-d tekivad erinevates ajapunktides (drop-out), siis kirjelda seda ajapunkti kaupa.
2. Mis võiks põhjustada andmete puudumist?
3. Kas mõõteobjektid, millel esineb NA-sid, erinevad süstemaatiliselt neist, millel ei ole NA-sid? Kas need grupid erinevad keskmise ja/või varieeruvuse poolest? Kuidas erineb *complete case* analüüsi tulemus imputeeritud regressioonist?
4. Imputatsiooni meetod, selle eeldused (MAR).
5. Software ja selle mitte-vaike settingud.
6. Mitu imputatsiooni tabeli kohta?
7. Imputatsiooni mudeli kirjeldus: millised muutujad, kas kasutati automaatset muutujate selektsiooni, kuidas imputeeriti mitte-normaalseid ja kategoorilisi muutujaid, kuidas võeti arvesse katse hierarhilist struktuuri?
8. Kuidas imputeeriti transformeeritud muutujaid (bmi, suhted jms)?
9. algse ja imputeeritud andmete võrdlus, konvergentsi diagnostika. Kas imputeeritud andmed on usutavad?

Punktid 1, 2, 4, 5 ja 6 on krutsiaalsed.

Appendix A

Bayesi ja sagedusliku statistika võrdlus

```
library(tidyverse)
library(rethinking)
library(ggthemes)
library(brms)
```

Kaks statistikat: ajaloost ja tõenäosusest

Bayesiaanlik ja sageduslik statistika leitati üksteise järel Pierre-Simon Laplace poolt, kes arendas välja kõigepealt bayesiaanliku statistika alused ning seejärel sagedusliku statistika omad (ca. 1774 - 1814). Sagedusliku statistika õitsengu põhjusteks 20. sajandil olid arvutuslik lihtsus ning tõenäosuse sagedusliku tõlgenduse sobivus 20 saj esimeses pooles käbinud teadusfilosoofiatega - eeskätt loogilise postivismiga. 1930-1980-ndatel valitses akadeemiliste statistikute seas seisukoht, et Bayesiaanlik statistika on surnud ja maha maetud, ning selle arendamisega tegelesid vaid üksikud inimesed, kes sageli olid füüsikaliste teaduste taustaga (Jeffreys, Jaynes).

Alates 1960-e keskpaigast arendati bayesiaanlust USA sõjaväe egiidi all, kuna seal oli piisav juurdepääs arvutivõimsusele, kuid seda tehti paljuski salastatult. Bayesi meetoditega ei olnud võimalik korralikult tsiviilteadust teha enne 1990-ndaid aastaid, mil personaalarvutite levik algatas buumi nende meetodite arendamises. Praegu on maailmas bayesiaanlikku ja sageduslikku statistikat umbes pooleks (vähemalt uute meetodite arendustöö poole pealt). Eestis bayesiaanlik statistika 2017 aasta seisuga peaaegu, et puudub.

1930ndatel kodifitseeris Andrei Kolmogorov tõenäosusteooria aksioonid (3 aksioomi), mis ütlevad lühidalt, et tõenäosused jäavad 0 ja 1 vahele ning, et üksteist välistavate ja hüpoteesiruumi ammendavate hüpoteeside tõenäosused summeeruvad ühele. Selgus, et Bayesi teoreem on lihtsa aritmeetika abil tuletatav Kolmogorovi aksioomidest. Tagantjärele saame öelda, et bayesiaanlik statistika on mitte ainult tõenäosusteooriga kooskõlas vaid ka, et Bayesi teoreem on parim võimalik viis sellist kooskõla saavutada (see on 1950ndate tarkus - Cox'i teoreem). On ka teada, et kui tõenäosused on fikseeritud nulli ja ühega, siis taandub Bayesi teoreem klassikalisele lausearvutuslikule loogikale. See tähendab, et klassikaline loogika on bayesiaanluse erijuht. Seevastu sageduslik statistika püüab saavutada mõistlike lahendusi arvutuslikult lihtsamate meetoditega, mille hinnaks on formaalse kooskõla puudumine tõenäosusteooriga. Seega kujutab sageduslik statistika endast kogumit *ad hoc* meetodeid, mis ei tähenda muidugi, et sellest kasu ei võiks olla. Küll aga tähendab see, et kuigi sageduslike mudeleid on lihtsam arvutada, on neid raskem ehitada ja mõista ning, et sageduslike testide, milliseid on viimase saja aasta jooksul loodud 10 000 ringis, tulemusi on raskem tõlgendada.

Kahe statistika põhilise erinevus ei tulene tõenäosusteooria matemaatikast, vaid erinevatest tõenäosuse tõlgendustest.

Bayesi tõlgenduses on tõenäosus teatlase usu määr mingi hüpoteesi kehtimisse. Hüpotees võib näiteks olla, et järgmise juuliku sademete hulk Vilsandil jäääb vahemikku 22 kuni 34 mm. Kui Bayesi arvutus annab selle hüpoteesi tõenäosuseks 0.57, siis oleme me selle teadmise ajal nõus maksma mitte rohkem kui 57 senti kihlveo eest, mille alusel makstakse juhul, kui see hüpotees töveseks osutub, välja 1 EUR (ja me saame vähemalt 43 senti kasumit).

Sageduslikud teoreetikud usuvad, et selline tõenäosuse tõlgendus on ebateaduslik, kuna see on "subjektiivne". Nimelt on võimalik, et n teadlast arvutavad samade andmete, kuid erinevate taustateadmiste põhjal n erinevat korrektset tõenäosust. Veelgi hullem, kui nad lähtuvad väga erinevatest taustauskumustest, ei pruugi toimuda teadmiste konvergeerumist ka siis, kui nende arvutustesse andmeid järjest juurde tuua.

Sellise olukorra osaline analoogia poliitikas on elanikkond, kus pooled kannavad konservatiivseid väärtsusi (perekond, rahvusühtsus) ja teine pool liberaalseid (multikultuursus, üldnimlikud väärtsused). Seega priorid on erinevad. Senikaua kui mõlemad pooled saavad oma uusides samast allikast (sama tõepära), ei ole demokraatia siiski ohus. Aga kui pooled hakkavad erinevatest allikatest hankima erinevaid fakte, mis kummagi ideoloogiat kinnitavad, toimub arvamuste polariseerumine ning demokraatia sattub ohtu.

Seega, kui te usute, et teie taustateadmised ei tohi mõjutada järeldusi, mis te oma andmete põhjal teete, siis te ei ole bayesiaan. Siinkohal pakub alternatiivi tõenäosuse sageduslik tõlgendus. Sageduslik tõenäosus on defineeritud kui teatud tüüpi andmete esinemise pikajaline suhteline sagedus. Näiteks, kui me viskame

münti palju kordi, siis peaks kullide (või kirjade) suhteline sagedus andma selle mündi tõenäosuse langeda kiri üleval. Selline tõenäosus on omistatav ainult sellistele sündmustele, mille esinemisel on sagedus. Kuna teaduslikul hüpoteesil ei ole esinemise sagedust, ei ole sageduslikus statistikas võimalik rääkida ka hüpoteesi kehtimise tõenäosusest. See on tõsine probleem, kuna tüüpiline statistiline hüpotees ütleb: meie mõõtmiste keskväärtus jäab vahemikku a kuni b – ja me tahaksime sellele anda usaldusintervallid, mis ütleksid, millise tõenäosusega see hüpotees kehtib. Sageduslik lahendus on selle asemel, et rääkida meie hüpoteesi tõenäosusest meie andmete korral, rääkida andmete, mis sarnanevad meie andmetega, esinemise tõenäosusest null-hüpoteesi (mis ei ole meie hüpotees) kehtimise korral. Seega omistatakse sagedus ehk tõenäosus andmetele, mitte hüpoteesile. Samas bayesiaan, kelle tõenäosused kehtivad hüpoteesidele, räägib otse tõenäosuse, millega parameetri väärthus jäab a ja b vahele, või ükskõik millisesse muusse teid huvitavasse vahemikku.

Poleemika: kumbki tõenäosus pole päris see, mida üldiselt arvatakse

Bayesi tõenäosus ei anna tegelikult seda tõenäosusnumbrit, mida me reaalselt peaksime kihlveokontoris kasutama. Ta annab numbri, millest me lähtuksime juhul, kui me usuksime, et selle numbri arvutamisel kasutatud statistilised mudelid kirjeldavad täpselt maailma. Paraku, kuna mudeldamine on oma olemuselt kompromiss mudeli lihtsuse ja ennustusvõime vahel, ei ole meil põhjust sellist asja uskuda. Seega ei peaks me bayesi tõenäosusi otse maailma üle kandma, vähemasti mitte automaatselt. Bayes ei ütle meile, mida me reaalselt usume. Ta ei ütle, mida me peaksime uskuma. Ta ütleb, mida me peaksime uskuma tingimuslikult.

Sageduslik tõenäosus on hoopis teine asi. Seda on võimalik vaadelda kahel viisil:

1. imaginaarsete andmete esinemissagedus nullhüpoteesi all;
2. reaalsete sündmuste esinemise sagedus.

Teise vaate kohaselt on sageduslik tõenäosus päriselt olemas. See on samasugune füüsikaline nähtus nagu näiteks auto kiirus, mõõdetuna liiklusmiilitsa poolt.

Kui kaks politseinikku mõõdavad sama auto kiirust ja 1. saab tulemuseks 81 km/h ning 2. saab 83 km/h, siis meie parim ennustus auto kiiruse kohta on 82 km/h. Kui aga 1. mõõtmistulemus on 80 km/h ja teine 120 km/h, siis meie parim hinnang ei ole 100 km/h. Enne sellise hinnangu andmist peame tegema lisatööd ja otsustama, kumb miilits oma mõõtmise kihva keeras. Ja me ei otsusta seda mitte oodatavast trahivist lähtuvalt, vaid neutraalseid objektiivseid asjaolusid vaagides. Seda sellepärast, et autol on päriselt kiirus olemas ja meil on hea põhjus, miks me tahame seda piisava täpsusega

teada. Sagedusliku statistiku mõõteriist on statistiline mudel ja mõõtmistulemus on tõenäosus, mis jäab 0 ja 1 vahele.

Õpikunäidetes on sündmusteks, mille esinemise sagedust tõenäosuse abil mõõdetakse, enamasti täringuvisked, ehk katsesüsteemi reaalne füüsikaline funktioneerimine. Pane tähele, et need on inimtekkelised sündmused (loodus – ega jumal – ei viska täringuid). Teaduses on sündmused, millele tõenäosusi omistatakse, samuti inimtekkelised: selleks sündmuseks on teadlase otsus H_0 ümber lükkamise kohta, mille tegemisel ta lähtub p (või q) väärthusest ja usaldusnivoost. Siin vastab auto kiirusele 1. tüüpi vigade tegemise sagedus. See sagedus on inimtekkeline, aga sellest hoolimata pärisele olemas ja objektiivselt mõõdetav. Kui 2 teadlast mõõdavad seda paraleelselt ja saavad piisavalt erineva tulemuse (näiteks väga erineva FDR-i), võib olla kindel, et vähemalt üks neist eksib, ning peaks olema võimalik ausalt otsustada, kumb.

Võrdlev näide: kahe gruvi võrdlus

Järgnevalt toome näite, kuidas bayesiaan ja sageduslik statistik lahendavad sama ülesande. Meil on 2 gruupi, katse ja kontroll, milles kummagis 30 mõõtmist ja me soovime teada, kui palju katsetingimus mõjutab mõõtmistulemust. Meie andmed on normaaljaotusega ja andmepunktid, mida me analüüsime, on efektisuurused (katse1 - kontroll1 = ES1 jne).

Bayesiaan

Statistiline küsimus on Bayesiaanil ja sageduslikul statistikul sama: kas ja kui palju erinevad kahe gruvi keskväärtused? Bayesiaan alustab sellest, et ehitab kaks mudelite: andmete tõepäramudel ja taustateadmiste mudel ehk prior.

Kui andmed on normaaljaotusega, siis on ka tõepäramudel normaaljaotus. Alustame sellest, et fitime oma valimiandmed (üksikud efekti suurused) normaaljaotuse mudelisse.

See ei ole veel tõepäramudel, sest me tahame hinnangut ES **keskväärtuse** kõige tõenäolisemale väärtsusele, ja lisaks veel hinnangut ebakindlusele selle punkt-hinnangu ümber (usalduslpiire). Seega tuleb eelmise jaotus kitsamaks tõmmata, et ta kajastaks meie teadmisi ES-ide keskväärtuste, mitte individuaalsete ES-de, kohta. Uue jaotusmudeli $sd = \text{eelmise jaotuse } sd / \sqrt{30}$.

Täpsemalt, selle joonise põhjal võib arvutada, milline on meie valimi keskväärtuse kohtamise tõenäosus igal võimalikul tõelisel ES-i väärtsel. Kõige tõenäolisemad on andmed siis, kui tegelik ES = andmete keskväärtusega (seda kohta näitab must joon). Kui me jagame musta joone pikkuse punase kurvi all läbi katkendjoone pikkusega sama kurvi all, saame teada, mitu korda on meie andmed tõenäolisemad siis, kui tegelik ES = mean(valimi ES), võrreldes olukorraga, kus

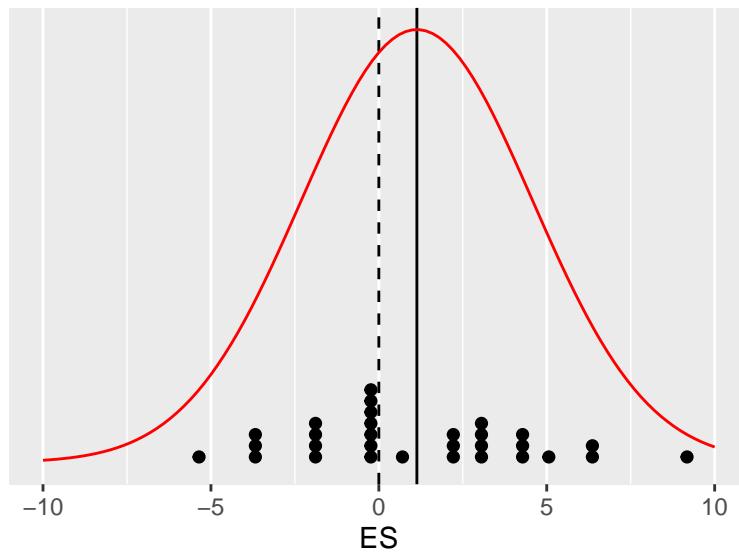


Figure A.1: Paariviisiline katse - kontroll disain. Katset on korratud 30 korda. X-teljel on efektisuurused (ES). 30 üksikut efektisuurust on näidatud punktidena. Must joon näitab keskmist efektisuurust. Andmed on mudeldatud normaaljaotusena.

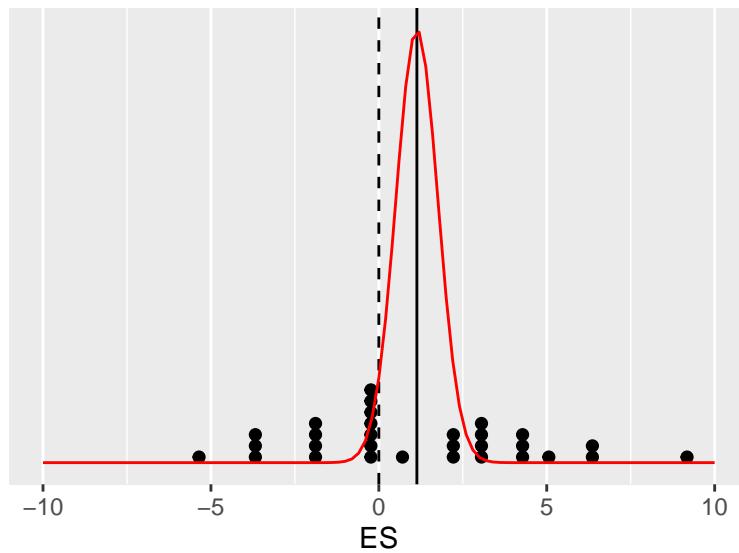


Figure A.2: See jaotus iseloomustab keskmise ES paiknemist puhtalt meie andmete põhjal.

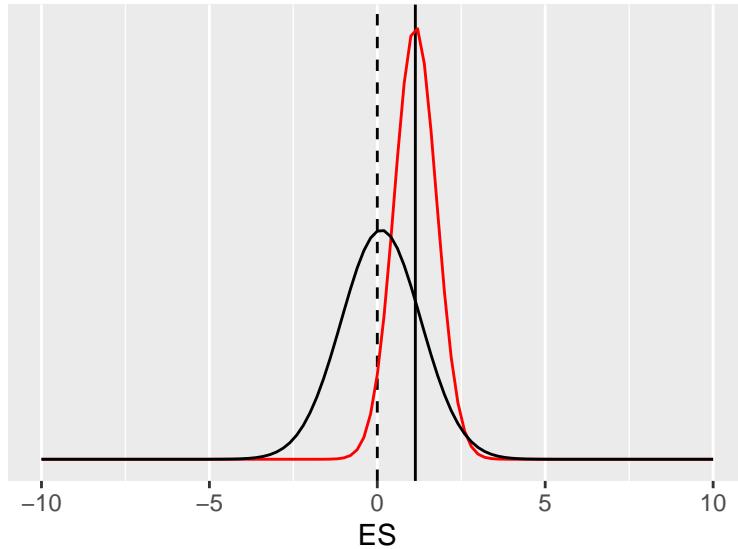


Figure A.3: Taustateadmiste mudel ehk prior on normaaljaotus (must joon), mille ülesanne on veidi vähendada ekstreemsete valimite kahjulikku mõju.

tegelik $ES = 0$. Loomulikult võime sama näitaja arvutada ükskõik millise hüpooteesi paari kohta (näiteks, andmed on miljon korda töenäolisemad hüpoteesi $ES = 0.02$ all kui hüpoteesi $ES = -1$ all; mis aga ei tähenda, et andmed oleksid väga töenäolised kummagi vörreldud hüpoteesi all).

Aga see ei ole veel Bayes. Lisame andmemudelile taustateadmiste mudeli. Sellega tühistame me väga olulise eelduse, mis ripub veskikivina sagedusliku statistika kaelas. Nimelt, et valimi andmed peavad olema esinduslikud populatsiooni suhtes. Me võime olla üsna kindlad, et väikeste valimite korral see eeldus ei kehti ja sellega seoses ei tööta ka sageduslik statistika viisil, milleks R.A. Fisher selle kunagi lõi. Taustateadmiste mudeli peamine, kuigi mitte ainus, roll on mõjutada meie hinnangut õiges suunas vähendades halbade andmete võimet meile kahju teha. Kui sul on väike valim, siis sinu andmed vajavad sellist kantseldamist.

Olgu meie taustateadmise mudel normaaljaotus keskväärtusega 0 ja standardhälbgaga 1.

Taustateadmiste mudel on sageli normaaljaotus. Kui meil on palju taustateadmisi, siis on see jaotus kõrge ja kitsas, kui meil on vähe taustateadmisi, siis on see madal ja lai.

Mida teha, kui sa ei taha, et taustateadmiste mudel sinu posteeriori kuju mõjutab? Sellisel juhul kasutatakse nõrgalt informatiivseid prioreid, mis tähendab, et priori jaotus on palju laiem kui töepäramudeli laius. Miks mitte kasutada mitte-informatiivseid tasaseid prioreid? Põhjused on arvutuslikud, seega tehnilist laadi.

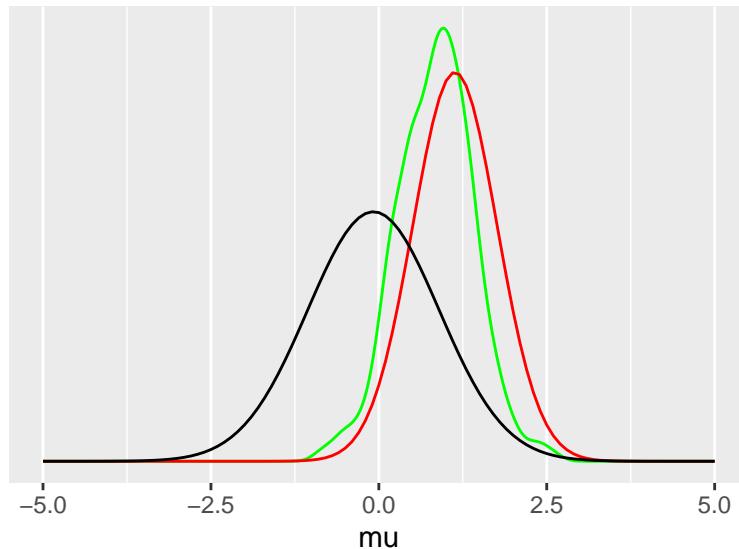


Figure A.4: Triplot. Bayesi väljund on posterioorne tõenäosusjaotus (roheline). Nagu näha, ei ole selle jaotuse tipp täpselt samas kohas kui andmejaotuse tipp ehk keskväärtus. Prior tömbab seda veidi nulli suunas. Lisaks on posteerior veidi kitsam kui andmemudel, mis tähendab, et hinnang ES-le tuleb väiksema ebakindluse määraga.

Igal juhul järgmisse sammuna korrutab bayesiaan selle jaotuse andmejaotusega, saades tulemuseks kolmanda normaaljaotuse, mille ta seejärel normaliseerib nii, et jaotuse alune pindala = 1. See kolmas jaotus on posterioorne tõenäosusjaotus, mis sisaldab kogu infot, millest saab arvutada kõige tõenäolisema katseefekti suuruse koos ebakindluse määraga selle ümber (mida rohkem andmeid, seda väiksem ebakindlus) ja tõenäosused, et tegelik katseefekt jääb üksköik milllisesse meid huvitavasse vahemikku.

Nüüd ei ole siis muud kui bayesi mudel läbi arvutada.

```
dfa <- data.frame(a)
m99 <- map2stan(
  alist(
    a ~ dnorm(mean = mu, sd = sigma),
    mu ~ dnorm(0, 1),
    sigma ~ dcauchy(0, 1)),
  data = dfa)
```

Posteerior sisaldab endas kogu infot, mis meil ES-i tõelise väärustuse kohta on. Siit saame arvutada:

1. parima hinnangu ES-i punktväärtusele,

2. usaldusintervalli, ehk millisest ES-ide vahemikust loodame leida tõelise ES-i näit 90% töenäosusega,
3. iga mõeldava ES-i väärustete vahemiku kohta töenäosuse, millega tõeline ES jäab sellesse vahemikku.
4. saame ES-i põhjal arvutada mõne muu statistiku, näiteks $ES_1 = \log(ES)$, kasutades selleks ES-i posterioorset jaotust. Sel viisil kanname oma ES-i hinnangus peituva ebakindluse üle ES_1 -le, millele saame samuti rakendada punkte 1-3 (sest ES_1 on posterioorne jaotus).
5. uute andmete lisandumisel saame kasutada ES-i posteeriorit uue priorina ja arvutada uue täiendatud posteeriori. Põhimõtteliselt võime seda teha pärast iga üksiku andmepunkti lisandumist. See avab ka head võimalused metaanalüüsiks.
6. lisaks saame oma algsest mudelist ka posteeriori andmepunkti tasemel varieeruvusele (pole näidatud). Seda kasutame uute andmete simuleerimiseks (meie näites üksikud ES-d).

Sageduslik statistik

Sageduslik lähenemine sisaldb ainult ühte mudelit, mida võrreldakse valimi andmetega. Sageduslik statistik alustab selles lihtsas näites täpselt samamoodi nagu bayesiaan, tekitudes eelmisega identse andmemudeli, mis on keskendatud valimi keskväärtusele B.2. Seejärel nihutab ta oma andmemudelit niipalju, et normaaljaotuse tipp ei ole enam valimi keskväärtuse kohal vaid hoopis 0-efekti kohal. Jaotuse laius nihutamisel ei muutu.

Seda nullile tsentreeritud mudelit kutsutakse null-hüpoteesiks (H_0). Nüüd võrdleb ta oma valimi keskväärtust (must joon) H_0 jaotusega. Kui valimi keskväärtuse kohal on H_0 jaotus kõrge, siis on andmete töenäosus H_0 kehitmise korral suur. Ja vastupidi, kui valimi keskväärtuse kohal on H_0 madal, siis on andmete esinemise töenäosus H_0 all madal. Seda töenäosust kutsutakse p vääruseks. Mida väiksem on p, seda vähem töenäolised on teie andmed juhul, kui H_0 on tõene ja katseefekt võrdub nulliga. P on defineeritud kui “teie andmete või 0-st veel kaugemal asuvate andmete esinemise pikaajaline suhteline sagedus tingimusel, et H_0 kehtib”.

Tulemuste tõlgendamine

Kui sageduslik statistik kirjutab, et tema “efekti suurus on statistiliselt oluline 0.05 olulisusnivool”, siis ta ütleb sellega, et tema poolt arvutatud $p < 0.05$. Selle väite korrektne tõlgendus on, et juhul kui statistik pika aja jooksul võtab omaks “statistikiliselt olulistena” kõik tulemused, millega kaasnev $p < 0.05$ ja lükkab tagasi kõik tulemused, mille $p > 0.05$, siis sooritab ta 5% sagedusega 1. tüüpi

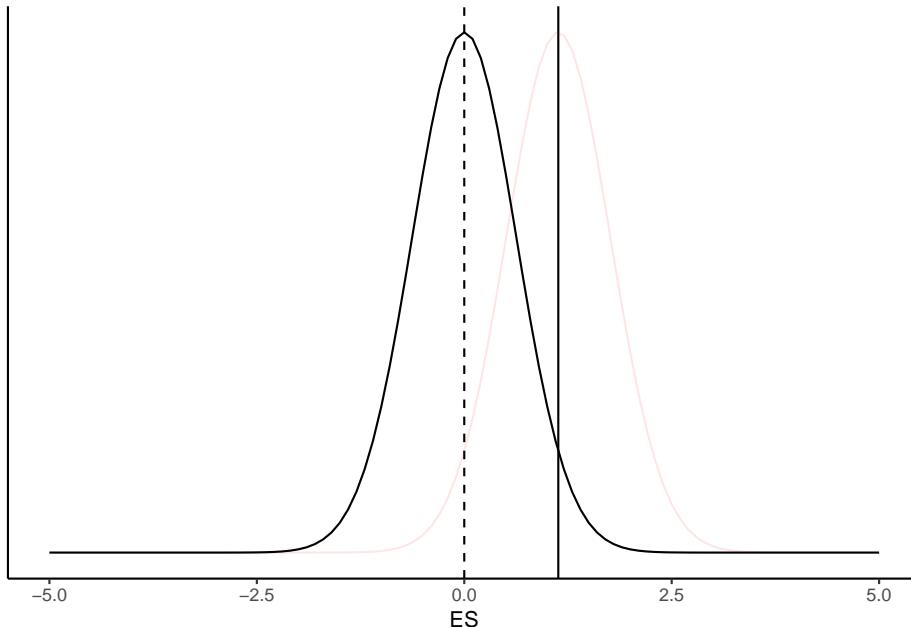


Figure A.5: Nullhüpotees (must kõver) ja tõepärafunktsioon (punane kõver).

vigu. See tähendab, et igast sajast tõesest H_0 -st, mida ta testib, võtab ta keskelt läbi 5 vastu, kui statistiliselt olulised. Sageduslik statistika on parim viis 1. tüüpi vigade sageuse fikseerimiseks.

Paraku ei tea me ühegi üksiku testi kohta ette, kas see testib kehtivat või mittekehtivat H_0 -i, mis teeb raskeks katseseeriate ühekaupa tõlgendamise. Tuletame meeal, et sagekuslikus statistikas ei saa rääkida H_0 kehtimise tõenäosusest vaid peab rääkima andmete tõenäosusest (ehk andmete esinemise sageusest) tingimusel, et H_0 kehtib.

Kas ühte p väärustust saab tõlgendada kui hinnangut tõendusmaterjali hulgale, mida teie valim pakub H_0 vastu? Selle üle on vaieldud juba üle 80 aasta, kuid tundub, et ainus viis seda kas või umbkaudu teha on bayesiaanlik. Igal juhul, p väärustust, mis on defineeritud pikaajalise sageusena, on raske rakendada üksiksündmusele. Bayesiaanliku p väärustuse tõlgendamiskalkulaatori leiate aadressilt <http://www.graphpad.com/quickcalcs/interpretPValue1/>.

Kujutle mass spektroskoopia katset, kus mõõdame 2000 valgu taseid katse-kontroll skeemis ja katset korrratakse n korda. Sageduslik statistik kasutab adjusteeritud p väärustusi või q väärustusi, et tõmmata piir, milles ühele poole jäavat statistiliselt olulised ES-d ja teiselle poole mitteolulised null-efektid. Edasi tõlgendab ta mitteolulisi efekte kui ebaolulisi ja diskuteerib vaid “olulisi” efekte. Paraku, p väärustute

arvutamine ja adjussteerimine saab toimuda mitmel erineval moel ja usalduspiiri panekule just 95-le protsendile, mitte näiteks 89% või 99.2%-le, pole ühtegi ratsionaalset põhjendust. Seega tõmbab ta sisuliselt juhuslikus kohas joone läbi efektide, misjärel ignoreerib kõiki sellest joonest valele poole jää nud efekte. Meetod, mis väga hästi töötab pikajalises kvaliteedikontrollis, ei ole kahjuks kuigi mõistlik katse tulemuste ükshaaval tõlgendamises. Mis juhtub, kui oleme kavalad ja proovime mitmeid erinevaid p väärustega töötamise meetodeid, et valida välja see usalduspiir, millega õigele poole jäävaid andmeid on teaduslikult kõige parem tõlgendada? Ehkki ükshaaval võisid kõik meie poolt läbi arvutatud meetodid olla lubatud (ja isegi vördselt head), ei fikseeri p nüüd enam 1. tüüpil vigade sagedust. See tähendab, et p on kaotanud definitsioonijärgse tähenduse ja te oleksite võinud olulisuspiiri sama hästi tõmmata tunde järgi.

Tüüpiline tulemuse kirjeldus artiklis:

1. sageduslik: *the effect is statistically significant ($p < 0.01$).*
2. bayesiaanlik: *the most likely effect size is xxx (90% CI = xxx-low, xxx-high) and the probability that the true effect is < 0 is xxx percent.*

90% CI — *credible interval* — tähendab, et me oleme 90% kindlad, et tegelik efekti suurus asub meie poolt antud vahemikus.

Kahe paradigma erinevused

1. sageduslikus statistikas võrdub punkt-hinnang tegelikule efekti suurusele valimi keskmise ES-ga. Bayesi statistikas see sageli nii ei ole, sest tausateadmiste mudel mõjutab seda hinnangut. Paljud mudelid püüavad ekstreemseid valimeid taustateadmiste abil veidi mõistlikus suunas niutada, niiviisi vähendades ülepaisutatud efektide avaldamise ohtu.
2. sageduslik statistika töötab tänu sellele, et uurija võtab vastu pluss-miinus otsuseid: iga $H \sim 0$ kas lükatakse ümber või jäetakse kehtima. Seevastu bayesiaan mötleb halli varjundites: sissetulevad andmed kas suurendavad või vähendavad hüpoteeside tõenäosust (mis jäavat aga alati > 0 ja < 1).
3. p vääritud kontrollivad 1. tüüpil vigade sagedust ainult siis, kui katse disaini ja hilisema tulemuste analüüs detailid on enne katse sooritamist fikseeritud (või eelnevalt on täpselt paika pandud lubatud variatsioonid katse- ja analüüs protokollis). Eelkõige tähendab see, et valimi suurus ja kasutatavad statistilised testid peavad olema eelnevalt fikseeritud. Tüüpiliselt saame p vääruse arvutada vaid üks kord ja kui $p = 0.051$, siis oleme sunnitud $H \sim 0$ paika jätma ning efekti deklareerimisest loobuma. Me ei saa lihtsalt katset juurde teha, et vaadata, mis juhtub. Bayesiaan seevastu võib oma posterioorse tõenäosuse arvutada kasvõi pärast iga

katsepunkti kogumist ning katse peatada kohe (või alles siis), kui ta leiab, et tema posterioorne jaotus on piisavalt kitsas, et teaduslikku huvi pakkuda.

4. Sageduslikus statistikas sõltub tulemus sellest, kas hüpotees, mida testitakse oli defineeritud enne andmete kogumist või mitte. Selle pärast tuleks seal testitavad hüpoteesid eel-registreerida. Bayesi statistikas pole aga vahet, kas hüpotees on formulieeritud enne või pärast andmete nägemist – seal loevad töepärafunktiooni loomisel ainult olemasolevad andmed (mitte andmed, mis oleksid võinud olla aga ei ole).
5. sagedusliku statistika pluss-miinus iseloom tingib selle, et kui tegelik efekti suurus on liiga väike, et sattuda õigele poole olulisusnivood, siis annavad statistiliselt olulisi tulemusi ülepaisutatud efektid, mida tekib tänu valimiveale. Nii saab süsteematiiliselt kallutatud teaduse. Bayesi statistikas seda probleemi ei esine, kuna otsused ei ole pluss-miinus tüüpi.
6. bayesi statistika ei fikseeri 1. tüüpi vigade sagedust. See-eest võitleb see nn valehäirete vastu, milleks kaasajal kasutatakse enim mitmetasemelisi *shrinkage* mudeliteid. See on bayesi vaste sageduslikus statistikas kasutatavatele mitmese testimise korrektsioonidele. Kui sageduslik statistik võitleb valehäiretega p väärtsi adjusteerides ja selle läbi olulisusnivood nihutades, siis bayesiaan kasutab *shrinkage* mudelit, et parandada hinnanguid üksikute efektide keskväärtustele ja nende sd-le, kasutades paindlikult kogu andmestikus leiduvat infot.

Sageduslik ja teaduslik hüpoteesitestimine

Teaduslik lähenemine töendusmaterjalile on sarnane kohtuliku uurimisega: me kogume töendusmaterjali senikaua, kuni oleme veendumud, et saame selle põhjal eelistada ühte hüpoteesi kõikide teiste arvelt. Seega, kui faktid meile piisavat surve avaldavad, võtame vastu pluss-miinus otsuse, et kohtualune süüdi või teaduslik hüpotees õigeks mõista. See otsus on meie tegevuse eesmärk ja meie tegevus oli suunatud selle eesmärgi täitmisele.

Sageduslik statistika võtab samuti vastu dihhotoomseid otsuseid, aga hoopis teisel moel. Seal ei ole otsus eesmärk, vaid vahend. Kui me lükkame tagasi teatud null hüpoteesid, aga mitte teised, saame me sellisel viisil fikseerida pikaajalise 1. tüüpi vigade sageduse. Me võtame vastu otsuseid, eesmärgiga tagada katsesüsteemi pikaajaline kvaliteet. Need otsused ei eelda, et me usuksime, et mõni konkreetne null hüpotees on tõene või väär ning matemaatilised protseduurid, mille alusel me neid otsuseid langetame, ei püüa määrata individuaalse null hüpoteesi tõelähedust või tõenäosust, et see H_0 ei kehti. Seega ei tähenda fakt, et me lükkasime ümber konkreetse nullhüpoteesi seda, et me usume, et see null hüpotees on väär.

H_0 -i ümber lükkamisel põhineva statistikaga kaasnevad järelmid, millest ehk olulisim on vajadus fikseerida andmete kogumise ja analüüsmeetodid enne, kui andmed on kogutud. See on nii tehnilikel põhjustel, mis on seotud puhtalt meie

sooviga fikseerida 1. tüüpi vigade sagedus. Sellise veasageduste fikseerimise hind on, et andmeanalüüs valikud ei saa sõltuda tegelikest andmetest (nende kvaliteedist, jaotusest jms). Siinkohal tuleb eraldi röhutada, et tegemist ei ole üldise teadusliku meetodi omadusega. Teaduses (ja bayesi statistikas) on mitte ainult täiesti normaalne vaid lausa vajalik vaadata andmeid kriitilise pilguga ja kujundada oma analüüs vastavalt andmete kvaliteedile. Ning kui andmed ei paku piisavalt töendusmaterjali, et me saaksime otsustada oma hüpoteesi kasuks või kahjuks, siis on igati mõistlik andmeid juurde korjata senikaua, kuni oleme veendunud ühte või teistpidi.

Oluline erinevus sagedusliku ja bayesi statistika vahel on, et kui sageduslik meetod fikseerib pikaajalise veasageduse aga ei arvuta üksikute hüpoteeside tõenäosust, siis bayesi meetod vastupidi arvutab üksikute hüpoteeside tõenäosused, aga ei fikseeri pikaajalisi veasagedusi. Kui meid ikkagi huvitavad veasagedused ja statistiline võimsus, saab neid ka bayesiaanlikult leida, arvutades oma mudeleid simuleeritud andmetega.

Statistiline ennustus kui mitmetasandiline protsess

Me võime vaadelda ennustavat statistikat mitmetasemelise protsessina, kus alusel tasemel on punkthinnang parameetri värtusele, selle pealoleval tasemel on hinnang ebakindlusele selle punkthinnangu ümber, ning 3. tasemel on omakorda hinnang ebakindlusele 2. taseme hinnangu ümber. Ja nii edasi lõpmatusse. Bayes erineb klassikalisest statistikast selle poolest, et kui Bayes ehitab 2. taseme hinnangu tõepära ja priori põhjal, siis klassikaline statistika kasutab selleks pelgalt tõepära (konverteerituna null hüpoteesiks). See on tähtis, kuna tõepära modelleerib ainult seda osa juhuslikust varieeruvusest punkthinnangu ümber, mida kutsutakse valimiveaks. Prior on võimeline arvesse võtma ka andmete kallutatust.

Kuna klassikalises statistikas ei ole formaalset priori mudelit, ei hindata klassikalised usaldusintervallid (2. tase) tõelist ebakindlust punktväärtuse ümber. Seda teevald bayesiaanlikud krediibiilsusintervallid, aga ainult siis, kui priorite koostamisse on tõsiselt suhtutud.

I Punkthinnang – enamasti aritmeetiline keskmene — modelleerib andmejaotuse tüüpilist elementti. Eeldus: me teame, milline on andmete jaotus.

II tõepärafunktsioon hindab ebakindlust punkthinnangu ümber. Modelleerib valimiviga, mis on seda suurem, mida vähem on teil andmeid. Eeldus 1: andmed on esinduslikud (andmejaotus = populatsiooni jaotus) Eeldus 2: mudel kirjeldab andmeid genereerivat mehhanismi (siit tulevad sageli lisaeeldused, nagu populatsiooni normaaljaotus, lineaarsus, sõltumatud sündmused valimi koostamisel, vigade sõltumatus, homoskedastilus jms)

III prior kohendab töepärafunktsiooni hinnangut Modelleerib (1) andmete esinduslikkust, mis on seda väiksem, mida väiksem on valim, ja (2) süstemaatilist viga. Eeldus: meil on oma andmetest sõltumatuid teadmisi populatsiooni jaotuse kohta

Kui valim on piisavalt suur, siis võime olla piisavalt kindlad, et andmed on esinduslikud ning klassikalise statistika hinnangud ebakindlusele punktväärtuse ümber muutuvad selle võrra usutavamaks.

Samas, sedamõõda kui valimi suurus kasvab, muutub töepärafunktsioon üha kitsamaks, mis tõstab omakorda tõenäosust, et tegelik parameetri väärthus jäab töepärafunktsiooni kõrgema osa alt välja, tingituna süstemaatilisest veast, mille suurus ei sõltu valimi suurusest. Seega töötab klassikaline statistika parimini keskmiselt suurte valimite (ja keskmiselt suure andmete varieeruvuse) korral.

A.0.1 Ajaloolist juttu: normaaljaotus, Bayes ja sageduslik statistika

(Anders Hald; A History of Parametric Statistical Inference from Bernoulli to Fisher, 1713-1935, Springer 2000)

Laplace sõnastas 1774. aastal statistiku tööpöllu järgmiselt: kirjeldamaks andmeid (vigade jaotust) tõenäosusfunktsioonina leia matemaatiline mudel, millel oleks lõplik arv parameetreid. Seejärel leia algoritm, mis minimeeriks vead meie hinnagutele nende parameetrite väärustele kohta. Seega oli eesmärk konverteerida andmete (mõõtmisvad) jaotus posterioorseks tõenäosusjaotuseks, mille pealt saaks omakorda arvutada usaldusintervallid meie hinnagu täpsusele.

Laplace, kes tegeles palju astronoomiliste mõõtmiste analüüsiga, lootis näidata, et mõõtmisandmete aritmeetiline keskmene on parim viis arvutada sellise posterioorse jaotuse kõige tõenäolisemat väärust (lokatsiooniparameetrit). Parim viis selles mõttes, et teoreetiliselt parima lokalisatsiooniparameetri ümber on võimalik arvutada kitsaimad veapiirid. Aritmeetilise keskmise selle pärast, et selle kasutamine oli laialt levinud, tundus intuitiivselt mõistlik ning oli arvutuslikult lihtne. Laplace probleemi lahendas Gauss ca. 1809, võttes kasutusele nii uue tõenäosusjaotuse – normaaljaotuse – kui ka uue lokatsiooniparameetri arvutusmeetodi – vähimruutude meetodi. Tema küsitud küsimus oli: millist jaotust ja hindamismeetodit oleks vaja kasutada, et lokatsiooniparameeter tuleks just aritmeetiline keskmene? Rõhutades normaaljaotuse tähtsust, leidis Laplace 1812. aastal, et köikidest sümmetrilistest andmejaotustest viib ainult normaaljaotus olukorrani, kus aritmeetiline keskmene kattub posterioorse jaotuse tipuga.

Kuna Laplace ei teadnud midagi tegelike veajaotuste kohta astronoomilistel mõõtmistel, oli tal raskusi andmejaotuse spetsifitseerimisega, mille pealt Bayesi teoreemiga posterioorne hinnanguvad jaotus arvutada. Mõõtmisvigu tavatseti mudeldada nelinurksete, kolmnurksete või kvadraat-jaotustega ja polnud ühtki teaduslikku põhjust eelistada üht jaotust teistele. See oli üks põhjuseid, miks Laplace hakkas arendama sageduslikku statistikat ja kasutas alates 1811 üha

vähem Bayesi teoreemi. (Alles 1818 näitas Bessel empiiriliselt, et astronoomilised mõõtmised on normaaljaotusega.) Teine põhjus hüljata Bayesi statistika oli seotud tehniliste raskustega priorite mudeldamisel, mistõttu Laplace oli sunnitud kasutama tasaseid prioreid, mis omistasid igale sündmusele/hüpoteesile võrdse eeltõenäosuse – ja oma ilmses absurduses tegid elu lihtsaks tema kriitikutele.

Sageduslik statistika kasutab oma alusena keskset piirteoreemi (Laplace 1810, 1812), mille kohaselt on paljude andmevalimite *aritmeetilised keskmised* normaaljaotusega, ja seda hoolimata andmete tegelikust jaotusest (eeldusel, et valimid on piisavalt suured; vt 6. ptk). Seega, senikaua kuni me ei modelleeri mitte ühe valimi empiirilist andmete jaotust (mida vajab Bayesi teoreem), vaid hoopis paljude virtuaalsete valimite pealt arvutatud *keskväärtuste jaotust*, ei pea me teadma, milline on andmete tegelik jaotus. Sellelt pinnalt ongi välja töötatud nullhüpoteesi testimine, millel põhineb suur osa 20. sajandi statistikast. 1908 näitas Edgeworth, et suurte valimite ja mitteinformatiivsete priorite korral annavad mõlemad meetodid (Bayes ja sageduslik) sama hinnangu parameetrväärtusele ja numbriliselt sama usaldusintervalli.

Sagedusliku statistika põhiprintsiibid: Fisher 1922.

Statistilisete meetodite eesmärk on andmete redutseerimine. Selleks on vaja vaadelda andmeid juhuvalimina hüpoteetilisest lõpmata suurest populatsioonist, mille jaotust saab kirjeldada suhteliselt väheste parameetritega mudeli abil. (Valimit iseloomustab “statistik”, populatsiooni aga “parameeter”)

Statistik puutub kokku kolme sorti probleemidega:

1. Spetsifikatsiooni probleemid kerkivad esile seoses populatsiooni jaotusmudeli spetsifitseerimisega.
2. Estimatsiooni probleemid kerkivad esile seoses algoritmidega, mille abil määratatakse hüpoteetilise populatsionimudeli parameetrite väärtsused.
3. Jaotuse probleemid kerkivad üles seoses valimite põhjal arvutatud statistikute jaotuste matemaatilise kirjeldamisega.

Heal estimatsioonil on omakorda kolm kriteeriumit:

1. Konsistentsus — statistik on kosnsistentne siis, kui arvutatuna lõpmata suurest valimist, mis võrdub populatsiooniga, tuleb selle statistiku väärtsus täpselt õige.
2. Efektiivsus — statistiku efektiivsus on suhe (protsent) selle sisemisest täpsusest võrrelduna teoreetiliselt efektiivseima statistikuga. Efektiivsus väljendab, kui suurt osa kogu kättesaadavast informatsioonist meie statistik kasutab. Efektiivsuse kriteerium: statistik, arvutatuna suurest valimist, annab vähima võimaliku standardhälbgaga normaaljaotuse.
3. Piisavus (sufficiency) — kriteerium: statistik on piisav kui ükski teine statistik, mida saab samast valimist arvutada, ei lisa informatsiooni hinnatava parameetri väärtsuse kohta.

Fisher poolt juurutatud sageduslik terminoloogia: parameeter, statistik, tõepära, dispersioon (variance; ANOVA), konsistentsus, efektiivsus, piisavus, informatsioon, null hüpotees, statistiline olulisus, p väärthus, olulisuse test, protsendipunkt, randomiseerimine, interaktsioon, faktoriaalne disain.

Tänu kesksele piirteoreemile on sageduslik statistika arvutuslikult palju lihtsam kui Bayesi statistika (arvutused saab teha paberil ja pliiatsiga), aga kuna me ei saa enam rääkida tegelike empiiriliste andmete jaotusest, vaid peame selle asemel kasutama lõpmata hulka virtuaalseid valimeid, ei saa me enam Bayesi teoreemi abil tõenäosusi pöörata (konverteerida meie andmete tõenäosus parameetriväärtuse x kehtimise korral, parameetriväärtuse x kehtimise tõenäosuseks meie andmete korral). Veelgi enam, me ei saa isegi rääkida meie andmete tõenäosusest parameetriväärtuse x kehtimise korral, vaid oleme sunnitud oma keelt ja meelt murdes rääkima "meie andmete või neist ekstreemsemate andmete pikajalisest suhtelisest sagedusest nullhüpoteesi (aga kahjuks mitte ühegi teise parameetriväärtuse) kehtimise korral" (Fisher ca. 1920). Nagu näha, on siin arvutuslikul lihtsuse sel kõrge kontseptuaalne hind.

Appendix B

Sõnastik

- Statistiline populatsioon (statistical population) – objektide kogum, millele soovime teha statistilist üldistust. Näiteks hinnata keskmist ravimi mõju patsiendipopulatsioonis. Või alkoholi dehüdrogenaasi keskmist Kcat-i.
- Valim (sample) – need objektid (patsiendid, ensüümiprepid), mida me reaalselt mõõdame.
- Juhuvalim (random sample) – valim, mille liikmed on populatsioonist valitud juhuslikult ja iseseisvalt. See tähendab, et kõigil populatsiooni liikmetel (kõikidel patsientidel või kõikidel võimalikel ensüümipreparatsioonidel) on võrdne võimalus sattuda valimisse JA, et valimisse juba sattunud liikme(te) põhjal ei ole võimalik ennustada järgmisena valimisse sattuvat liiget. Juhuvalim muudab lihtsamaks normaaljaotuse mudeli kasutamise bayesiaanlikes arvutustes, aga ta ei ole seal selleks absoluutsest vajalik. Seevastu pea kogu sageduslik statistika põhineb juhuvalimitel.
- Esinduslik valim (representative sample) – Valim on esinduslik, kui ta peegeldab hästi statistilist populatsiooni. Ka juhuvalim ei pruugi olla esinduslik (juhuslikult).
- valimiviga (sampling error, sampling effect) - määr, millega juhuvalimi põhjal arvutatud statistiku väärtus (näit keskväärtus) erineb populatsiooni parameetri väärtestest. valimiviga kutsutakse sageli ka juhuslikuks müraks.
- kallutatus e süsteematiiline viga (bias) - see osa statistiku väärtsuse erinevusest katsettingimuse ja kontrolltingimuse vahel, mis on põhjustatud millegi muu poolt, kui deklareeritud katse-interventsioon.
- Statistik (statistic) – midagi, mis on täpselt arvutatud valimi põhjal (näiteks pikkuste keskmine)
- Parameeter (parameter) – teadmata suurus populatsiooni tasemel, mille täpset väärust me saame umbkaudu ennustada, aga mitte kunagi täpselt

teada. Näiteks mudeli intercept, populatsiooni keskmine pikkus.

- Efekti suurus (effect size) - siin võrdub katsegrupi keskmine – kontrollgrupi keskmine. Leidub ka teistsuguseid es mõõte, milles Levinuim on Coheni d.
- standardhälve
- mad
- variatsiooni koefitsient
- Statistikiline mudel (statistical model) – matemaatiline formaliseering, mis koosneb 2st osast: deterministlik protsessi-mudel pluss juhuslik vea/varieeruvuse-mudel. Protsessi-mudeli näiteks kujutle, et mõõdad mitme inimese pikkust (x muutuja) ja kaalu (y muutuja). Sirge võrrandiga $y = a + bx$ (kaal = $a + b * \text{pikkus}$) saab anda determinismliku lineaarse ennustuse kaalu kohta: kui x (pikkus) muutub ühe ühiku (cm) võrra, siis muutub y (kaal) väärthus keskmiselt b ühiku (kg) võrra. Seevastu varieeruvuse-mudel on tõenäosusjaotus (näit normaaljaotus). Selle abil modelleeritakse y -suunalist andmete varieeruvust igal x väärthusel (näiteks, milline on 182 cm pikkuste inimeste oodatav kaalujaotus). Mudel on seega tõenäosuslik: me saame näiteks küsida: millise tõenäosusega kaalub 182 cm pikkune inimene üle 100 kilo. Mida laiem on varieeruvuse mudeli y -i suunaline jaotus igal x -i väärthusel, seda kehvemini ennustab mudel, millist y väärust võime konkreetelt oodata mingi x -i vääruse korral. Lineaarsete mudelite eesmärk ei ole siiski mitte niivörd uute andmete ennustamine (seda teevald paremini keerulised mudelid), vaid mudeli struktuurist lähtuvalt põhjuslike hüpoteeside püstitamine/kontrollimine (kas inimese pikkus võiks otseselt reguleerida/kontrollida tema kaalu?). Kuna selline viis teadust teha töötab üksnes lihtsate mudelite korral, on enamkasutatud statistilised mudelid taotluslikult lihtsustavad ja ei pretendeeri töelähedusele.
 - tõepära (likelihood)
 - prior e eeljaotus
 - posteerior e järeljaotus (posterior)
 - Tehniline replikatsioon (technical replication) – sama proovi (patsienti, ensüümipreparatsiooni, hiire pesakonna liiget) mõõdetakse mitu korda. Mõõdab tehnilik varieeruvust ehk mõõtmisviga. Seda püüame kontrollida parandades mõõtmisaparatuuri või protokolle.
 - Bioloogiline replikatsioon (biological replication) – erinevaid patsiente, ensüümipreppe, erinevate hiirepesakondade liikmeid mõõdetakse, igaüht üks kord. Eesmärk on mõõta bioloogilist varieeruvust, mis tuleneb mõõteobjektide reaalsetest erinevustest: iga patsient ja iga ensüümimolekul on erinev kõigist teistest omasugustest. Bioloogiline varieeruvus on teaduslikult huvitav ja seda saab visualiseerida algandmete tasemel (mitte keskvääruse tasemel) näiteks histogrammina. Teaduslikke järelusi tehakse bioloogiliste

replikaatide põhjal. Tehnilised replikaadid seevastu kalibreerivad mõõtesüsteemi täpsust. Kui te uurite soolekepikest E. coli, ei saa te teha formaalset järeldust kõigi bakterite kohta. Samamoodi, kui te uurite vaid ühe hiripesakonna/puuri liikmeid, ei saa te teha järeldusi kõikide hirite kohta. Kui teie katseskeem sisaldb nii tehnilisi kui bioloogilisi replikaate on lihtsaim viis neid andmeid analüüsida kõigepealt keskmistada üle tehniliste replikaatide ning seejärel kasutada saadud keskmisi edasistes arvutustes üle bioloogiliste replikaatide (näiteks arvutada nende pealt uue keskmise, standardhälve ja/või usaldusintervalli). Selline kahe-etapiline arvutuskäik ei ole siiski optimaalne. Optimaalne, kuid keerukam, on panna mõlemat tüüpia andmed ühte hierarhilisse mudelisse.

Tõenäosuse (P) reeglid on ühised kogu statistikale:

- P jääb 0 ja 1 vahele; $P(A) = 1$ tähendab, et sündmus A toimub kindlasti.
- kui sündmused A ja B on üksteist välistavad, siis tõenäosus, et toimub sündmus A või sündmus B on nende kahe sündmuse tõenäosuste summa — $P(A \vee B) = P(A) + P(B)$.
- Kui A ja B ei ole üksteist välistavad, siis $P(A \vee B) = P(A) + P(B) - P(A \& B)$.
- kui A ja B on üksteisest sõltumatud (A toimumise järgi ei saa ennustada B toimumist ja vastupidi) siis tõenäosus, et toimuvald mõlemad sündmused on nende sündmuste tõenäosuste korrutis — $P(A \& B) = P(A) \times P(B)$.
- Kui B on loogiliselt A alamosa, siis $P(B) < P(A)$
- $P(A | B)$ — tinglik tõenäosus. Sündmuse A tõenäosus, juhul kui peaks toimuma sündmus B. $P(\text{vihm} | \text{pilves ilm})$ ei ole sama, mis $P(\text{pilves ilm} | \text{vihm})$.
- Juhul kui $P(B) > 0$, siis $P(A | B) = P(A \& B) / P(B)$ ehk
- $P(A | B) = P(A) \times P(B | A) / P(B)$ — Bayesi teoreem.

Kuigi kõik statistikud lähtuvad tõenäosustega töötamisel täpselt samadest matemaatilistest reeglitest, tõlgendavad erinevad koolkonnad saadud numbreid erinevalt. Kaks põhilist koolkonda on sageduslikud statistikud ja Bayesiaanid.

- Tõenäosus, Bayesi tõlgendus (Bayesian probability) – usu määr mingisse hüpoteesi. Näiteks 62% tõenäosus (et populatsiooni keskmise pikkus < 180 cm) tähendab, et sa oled ratsionaalse olendina nõus kulutama mitte rohkem kui 62 senti kihlveo peale, mis võidu korral toob sulle sisse 1 EUR (ja 38 senti kasumit). Bayesi tõenäosus omistatakse statistilisele hüpoteesile (näiteks, et ravimiefekti suurus jääb vahemikku a kuni b), tingimusel, et sul on täpselt need andmed, mis sul on; ehk $P(\text{hüpotees} | \text{andmed})$.
- Tõenäosus, sageduslik tõlgendus (Frequentist probability) – pikaajaline sündmuste suhteline sagedus. Näiteks 6-te sagedus paljudel täringuvistel. Sageduslik tõenäosus on teatud tüüpia andmete sagedus, tingimusel et nullhüpotees (H_0) kehtib; ehk $P(\text{andmed} | H_0)$. Nullhüpotees ütleb enamasti, et uuritava parameetri (näiteks ravimiefekti suurus) väärthus on null. Seega, kui P on väike, ei ole seda tüüpia andmed kooskõlas arvamusega, et

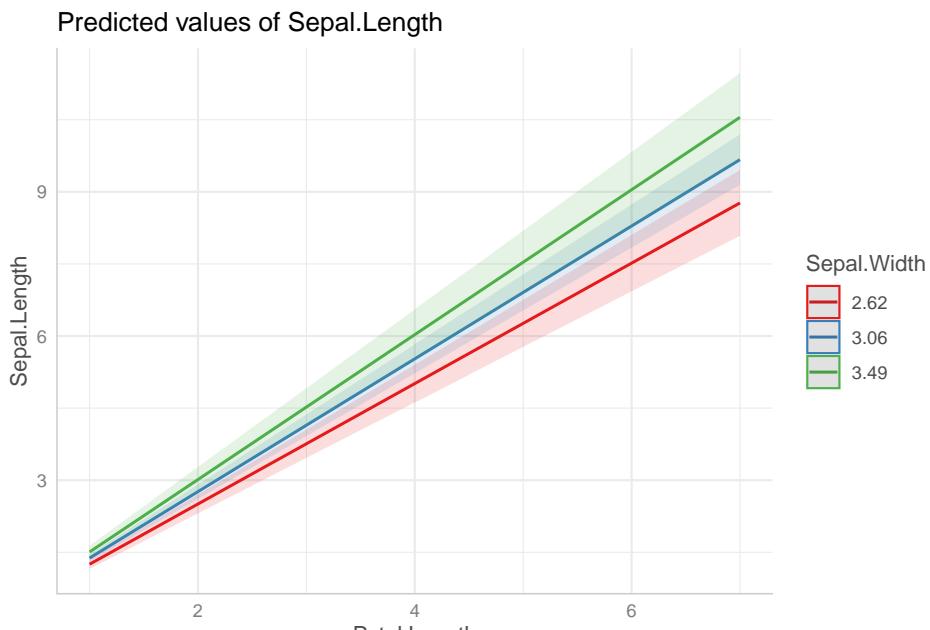
parameetri väärthus on null (mis aga ei tähenda automaatselt, et sa peaksid uskuma, et parameetri väärthus ei ole null).

```
aa
#library(tufte)
library(tidyverse)
library(ggthemes)
#library(brms)
library(broom)

library(ggeffects)
library(tidyverse)
library(broom)
```

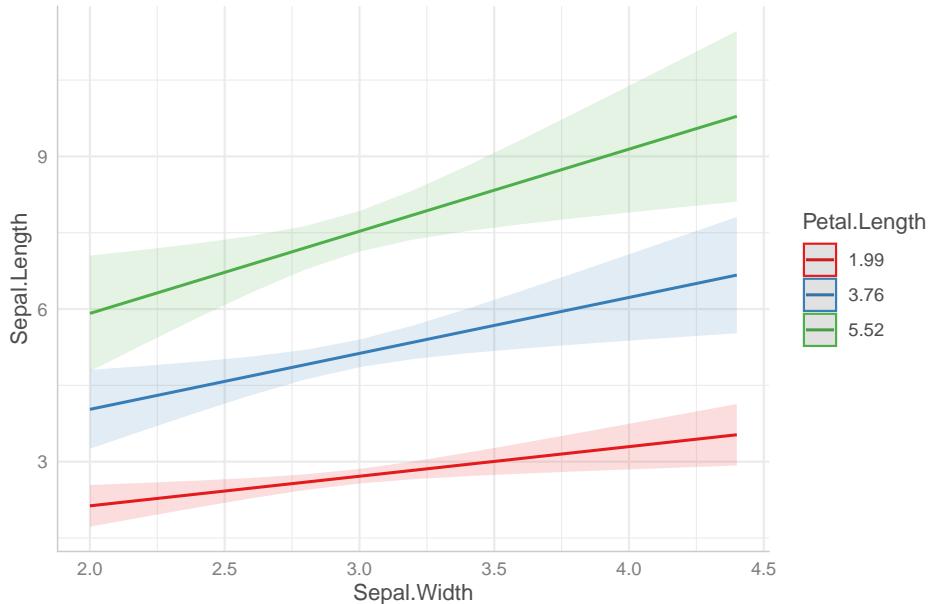
intercept surutud nulli, Sepal.Width-i tõus ei sõltu enam Petal.Lengthist.

```
m7 <- lm(Sepal.Length ~ 0 + Petal.Length + Petal.Length:Sepal.Width, data = iris)
plot(ggeffect(m7, terms=c("Petal.Length", "Sepal.Width")))
```



```
plot(ggeffect(m7, terms=c("Sepal.Width", "Petal.Length")))
```

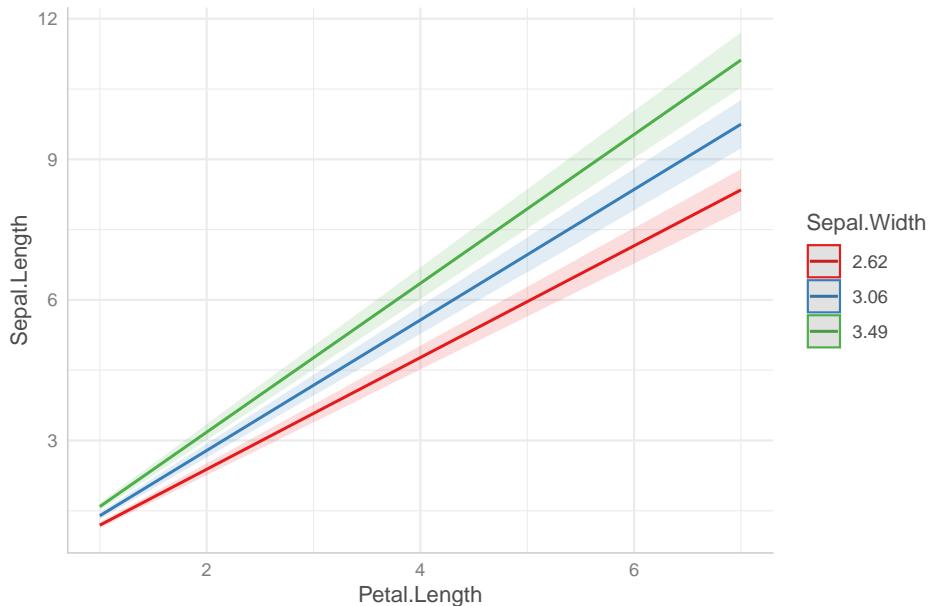
Predicted values of Sepal.Length



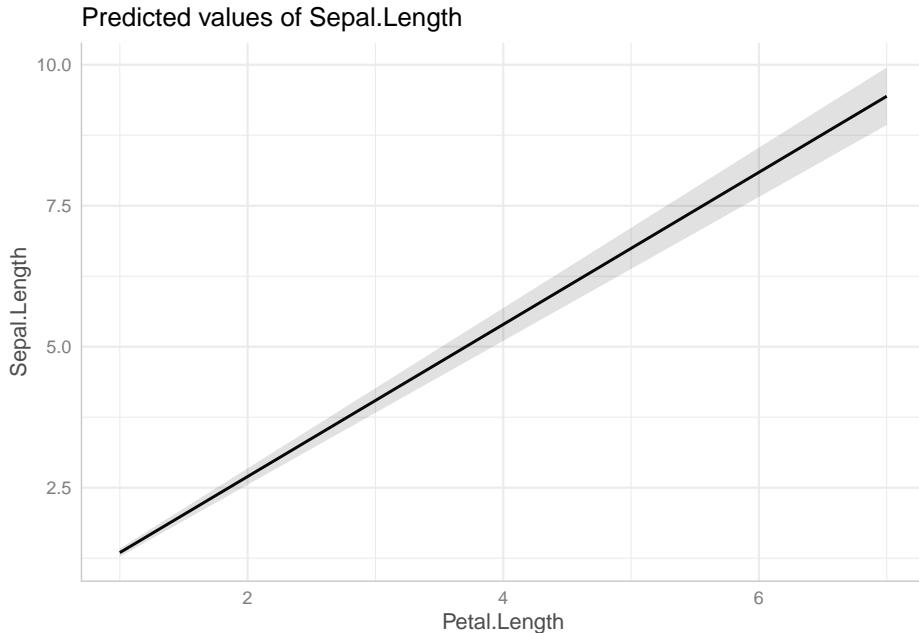
intercept surutud nulli. Me eeldame, et ei sepal width-l ega petal lengthil pole iseseisvat mõju y-le.

```
m8 <- lm(Sepal.Length ~ 0 + Petal.Length:Sepal.Width, data = iris)
plot(ggeffect(m8, terms=c("Petal.Length", "Sepal.Width")))
```

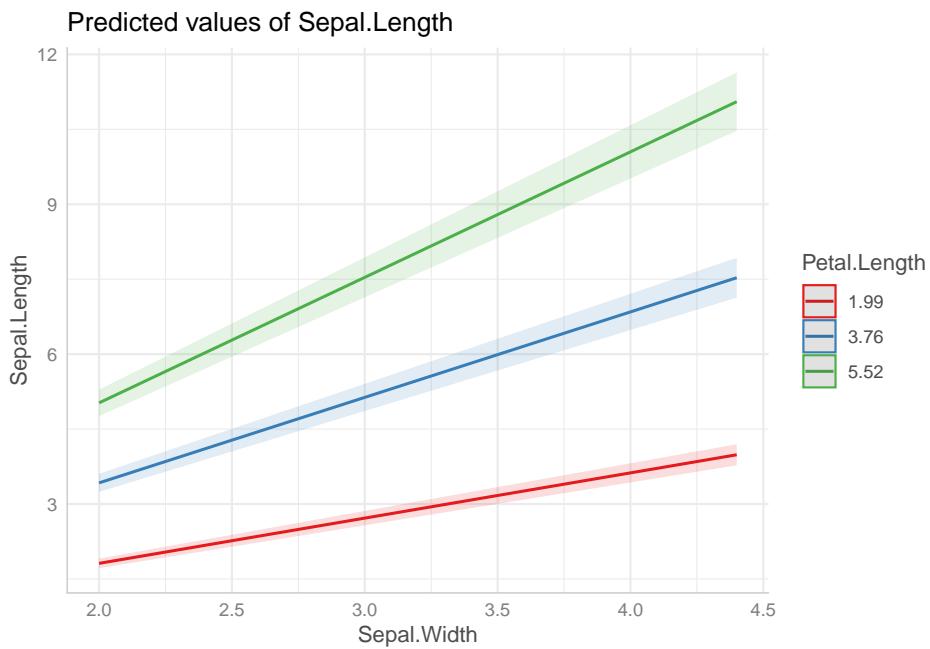
Predicted values of Sepal.Length



```
m8.1 <- lm(Sepal.Length ~ 0 + Petal.Length, data = iris)
plot(ggeffect(m8.1, terms=c("Petal.Length")))
```

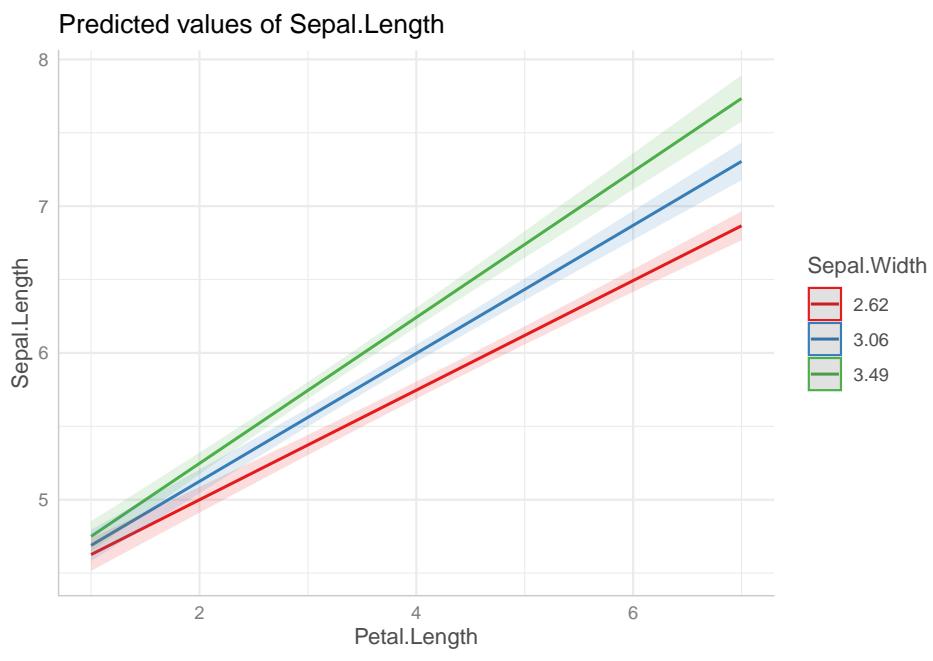


```
plot(ggeffect(m8, terms=c("Sepal.Width", "Petal.Length")))
```

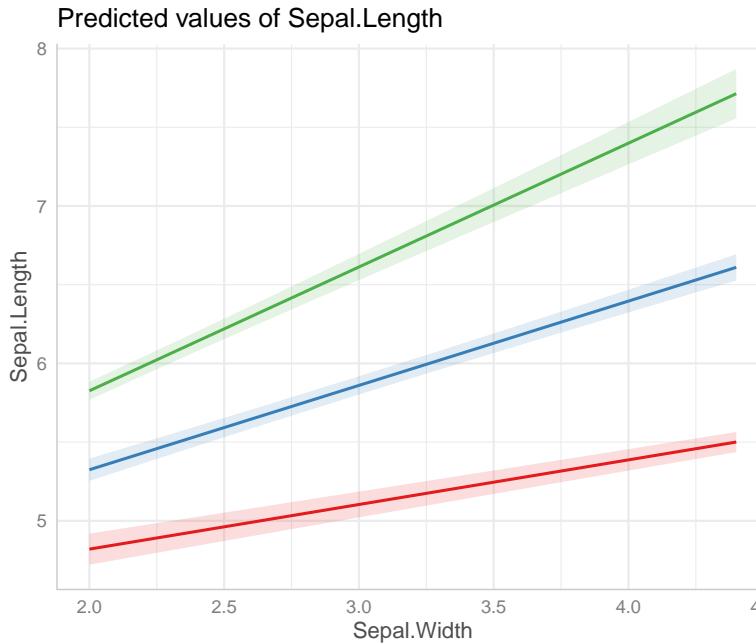


intercept fititakse, aga kummagil X-muutujal pole iseseisvat mõju y-le

```
m9 <- lm(Sepal.Length ~ 1 + Petal.Length : Sepal.Width, data = iris)
plot(ggeffect(m9, terms=c("Petal.Length", "Sepal.Width")))
```

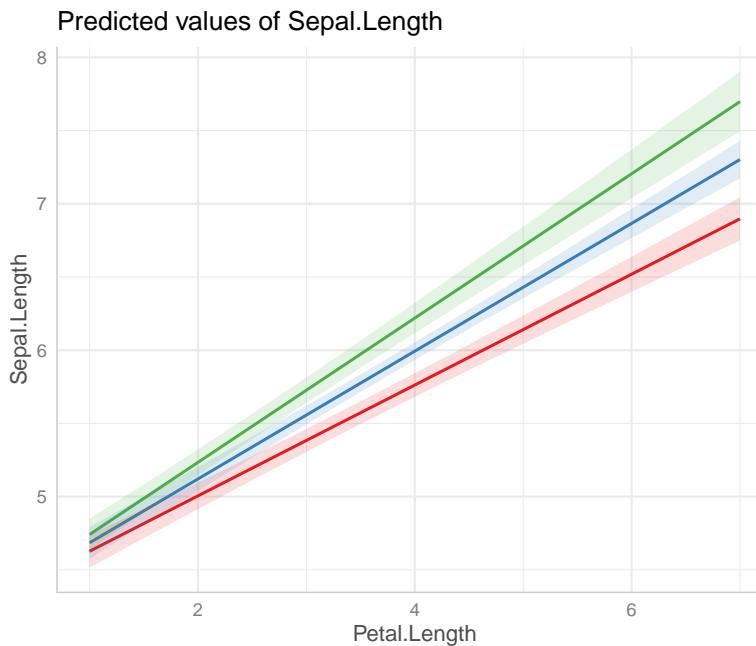


```
plot(ggeffect(m9, terms=c("Sepal.Width", "Petal.Length")))
```

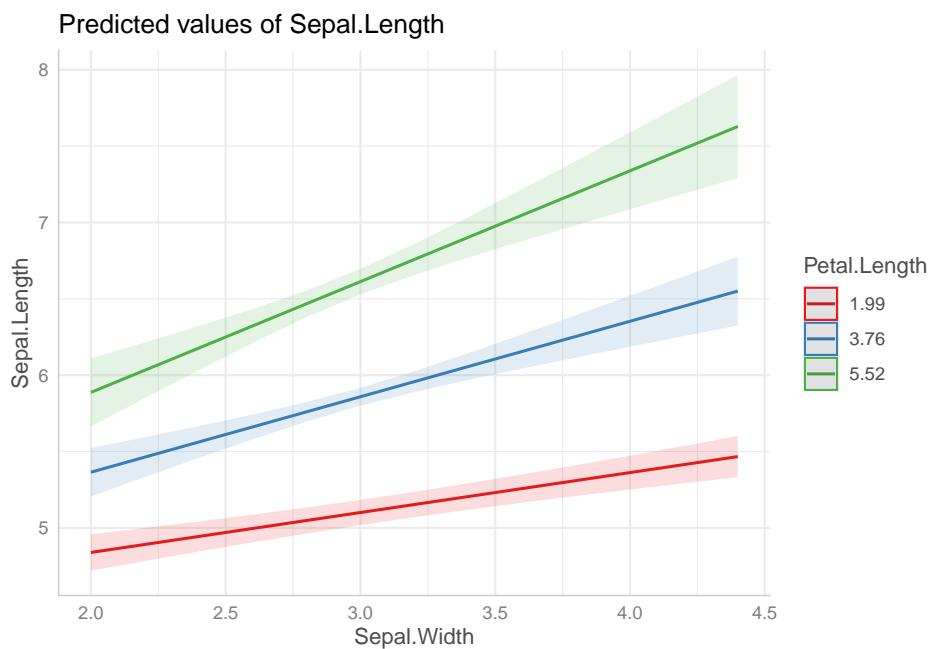


sepal width-il pole iseseisvat mõju y-le. kogu tema mõju on petal lengthi kaudu.

```
m10 <- lm(Sepal.Length ~ 1 + Petal.Length + Petal.Length:Sepal.Width, data = iris)
plot(ggeffect(m10, terms=c("Petal.Length", "Sepal.Width")))
```



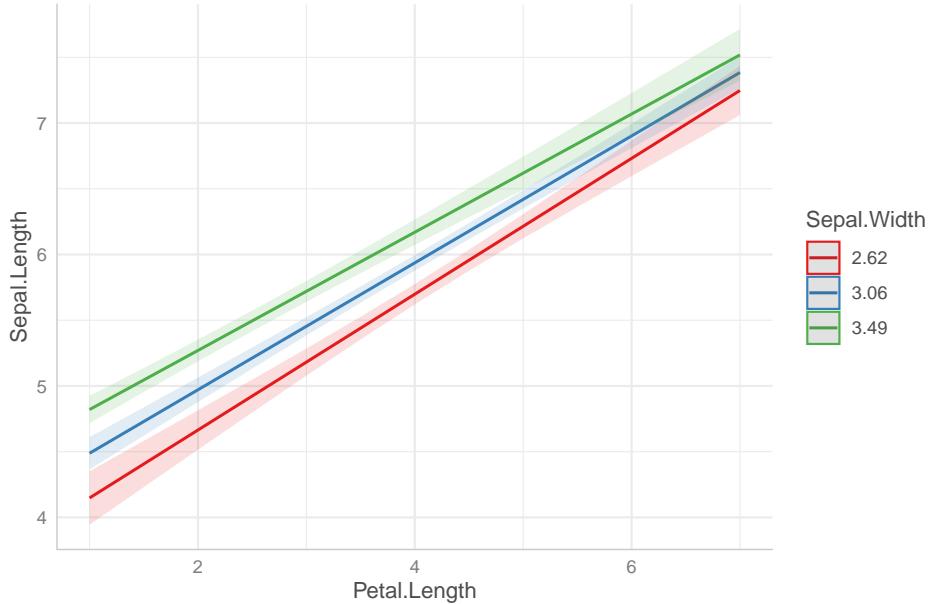
```
plot(ggeffect(m10, terms=c("Sepal.Width", "Petal.Length")))
```



täismudel.

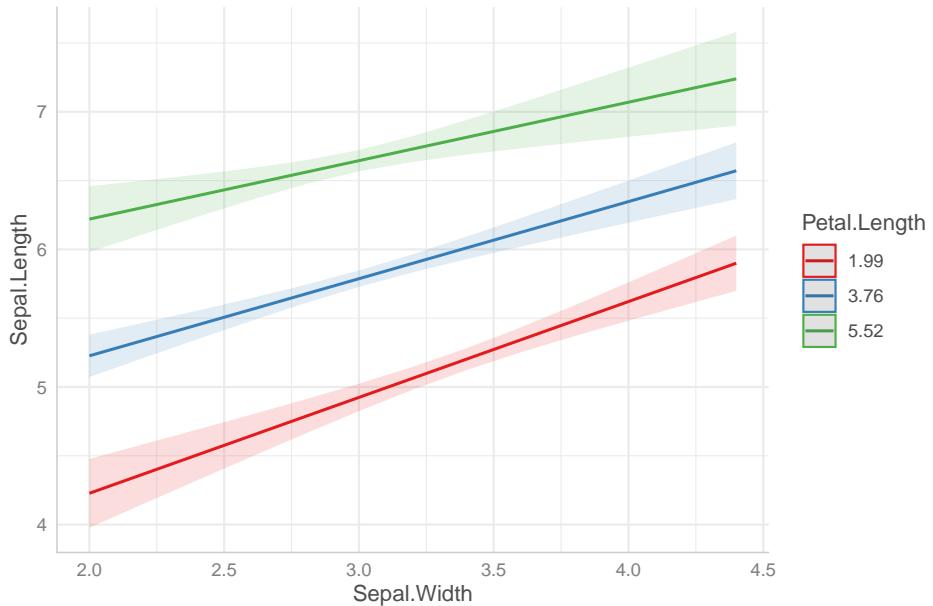
```
m11 <- lm(Sepal.Length ~ Petal.Length * Sepal.Width, data = iris)
plot(ggeffect(m11, terms=c("Petal.Length", "Sepal.Width")))
```

Predicted values of Sepal.Length



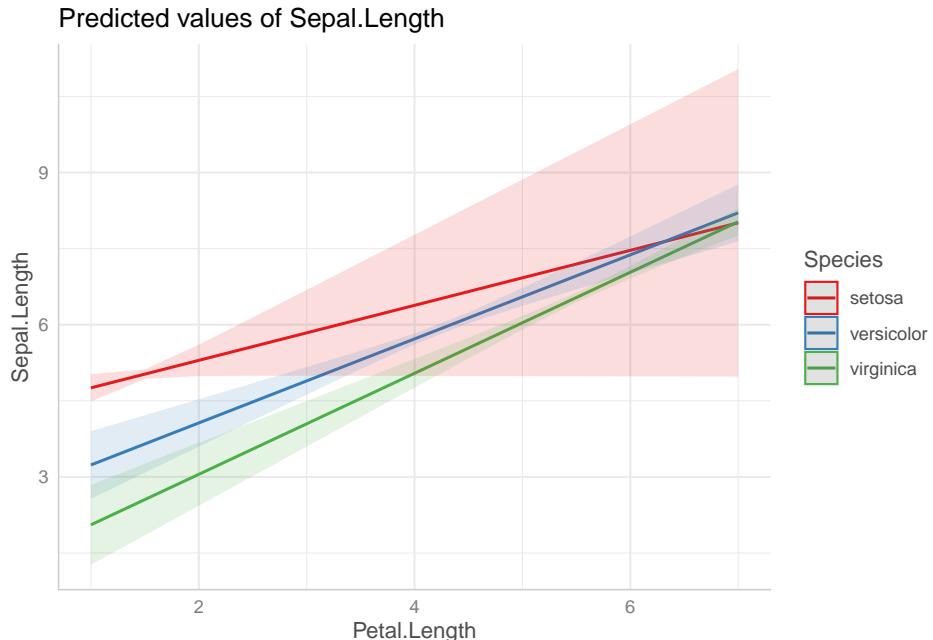
```
plot(ggeffect(m11, terms=c("Sepal.Width", "Petal.Length")))
```

Predicted values of Sepal.Length

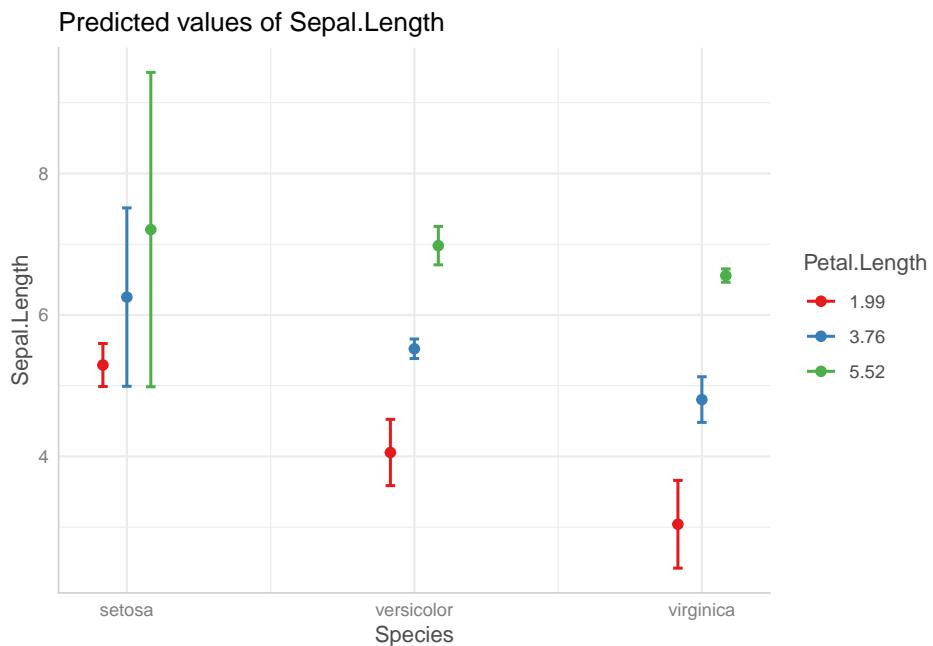


täismudel

```
m1 <- lm(Sepal.Length ~ Petal.Length * Species, data = iris)
plot(ggeffect(m1, terms=c("Petal.Length", "Species")))
```

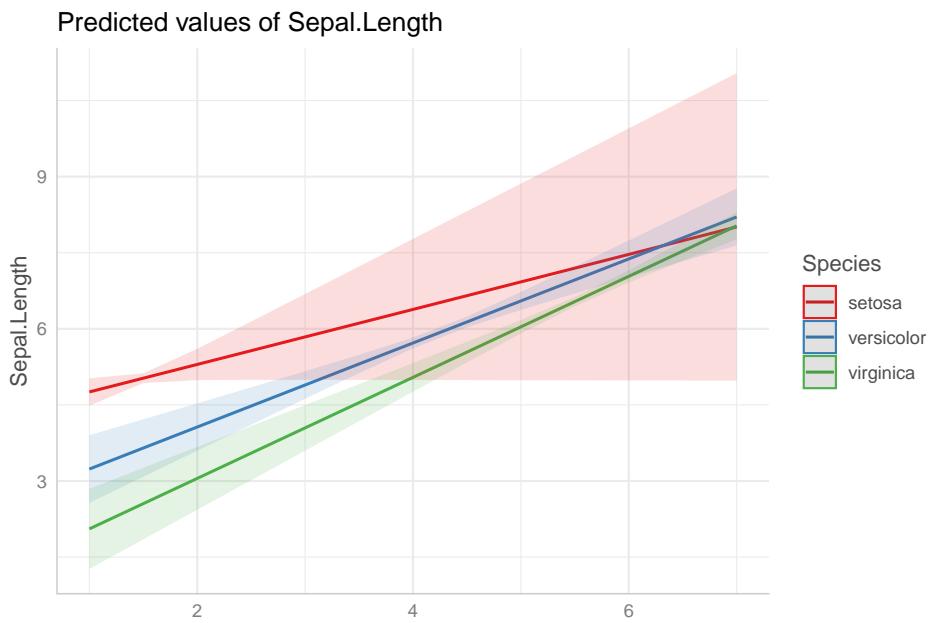


```
plot(ggeffect(m1, terms=c("Species", "Petal.Length")))
```



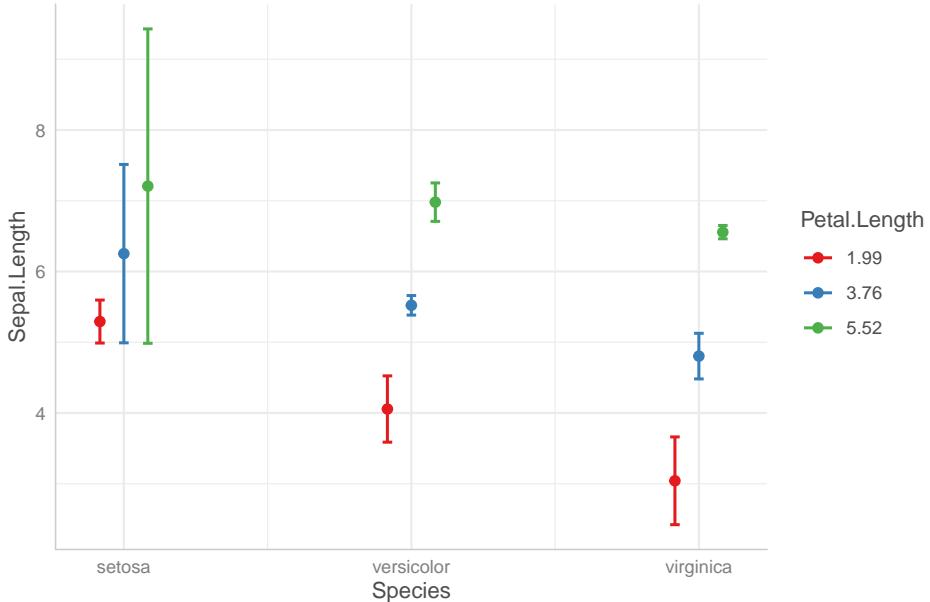
petal length is not allowed to have direct influence on y??? mudel paistab identne täismudeliga.

```
m2 <- lm(Sepal.Length ~ Species + Petal.Length:Species, data = iris)
plot(ggeffect(m2, terms=c("Petal.Length", "Species")))
```



```
plot(ggeffect(m2, terms=c("Species", "Petal.Length")))
```

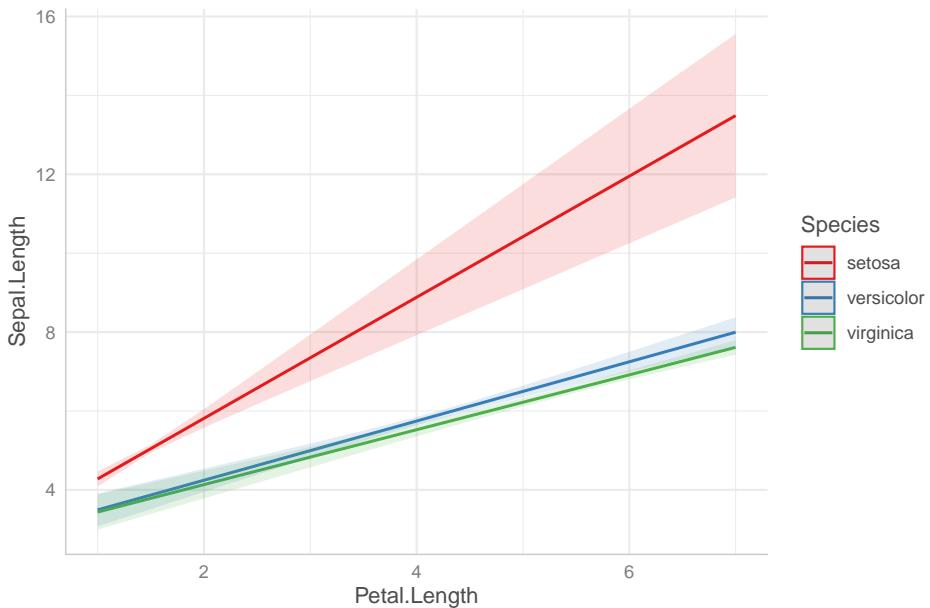
Predicted values of Sepal.Length



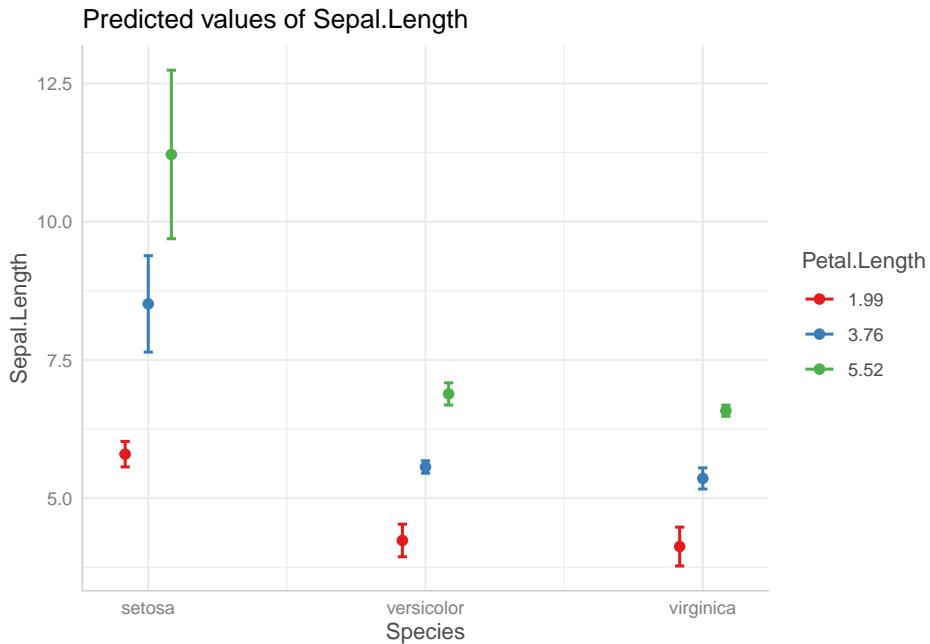
species ei ole lubatud omama ottest mõju y-le.

```
m3 <- lm(Sepal.Length ~ Petal.Length + Petal.Length:Species, data = iris)
plot(ggeffect(m3, terms=c("Petal.Length", "Species")))
```

Predicted values of Sepal.Length



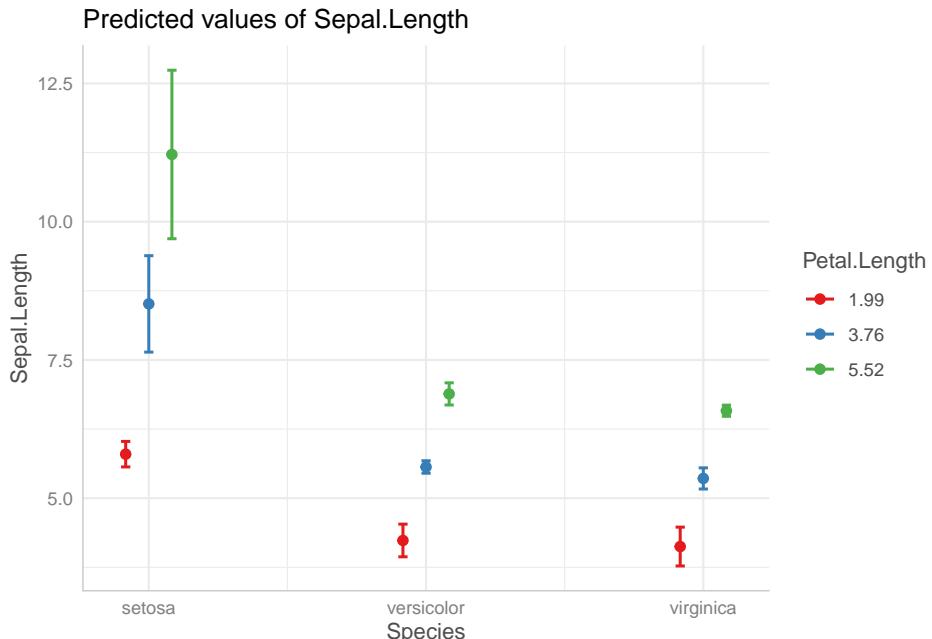
```
plot(ggeffect(m3, terms=c("Species", "Petal.Length")))
```



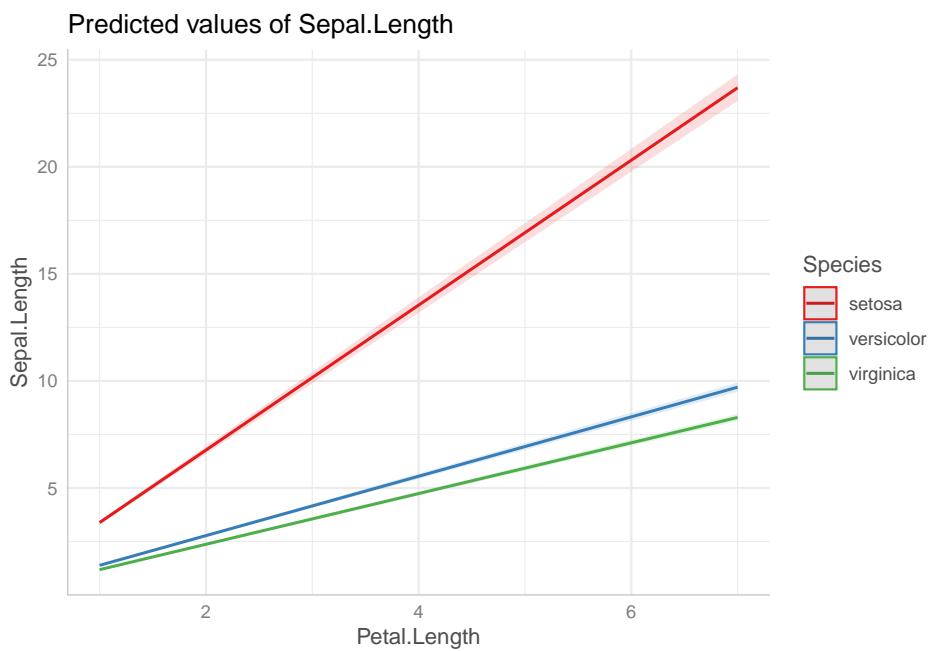
```
m4 <- lm(Sepal.Length ~ Petal.Length : Species, data = iris)
plot(ggeffect(m4, terms=c("Petal.Length", "Species")))
```



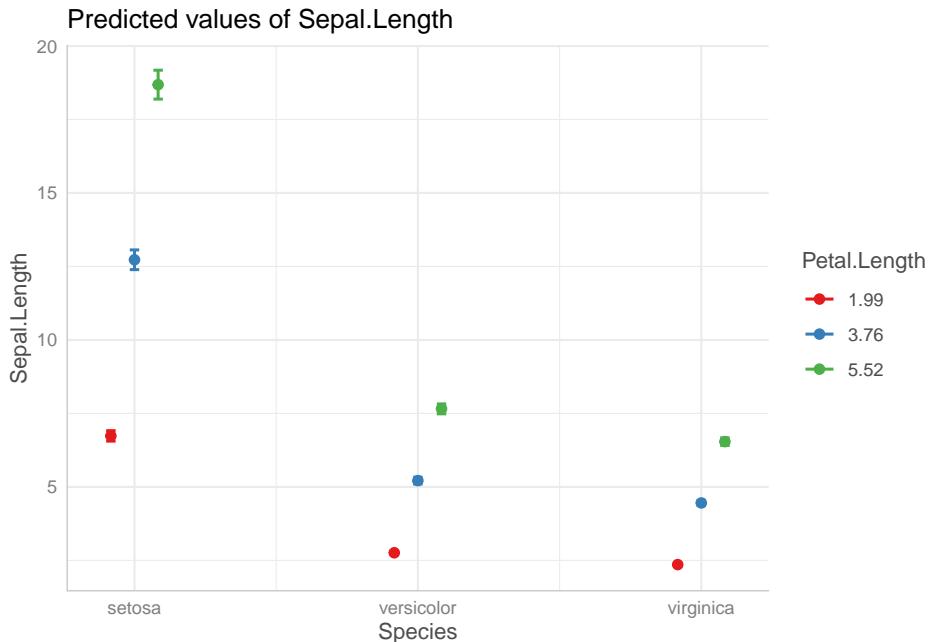
```
plot(ggeffect(m4, terms=c("Species", "Petal.Length")))
```



```
m5 <- lm(Sepal.Length ~ 0 + Petal.Length:Species, data = iris)
plot(ggeffect(m5, terms=c("Petal.Length", "Species")))
```



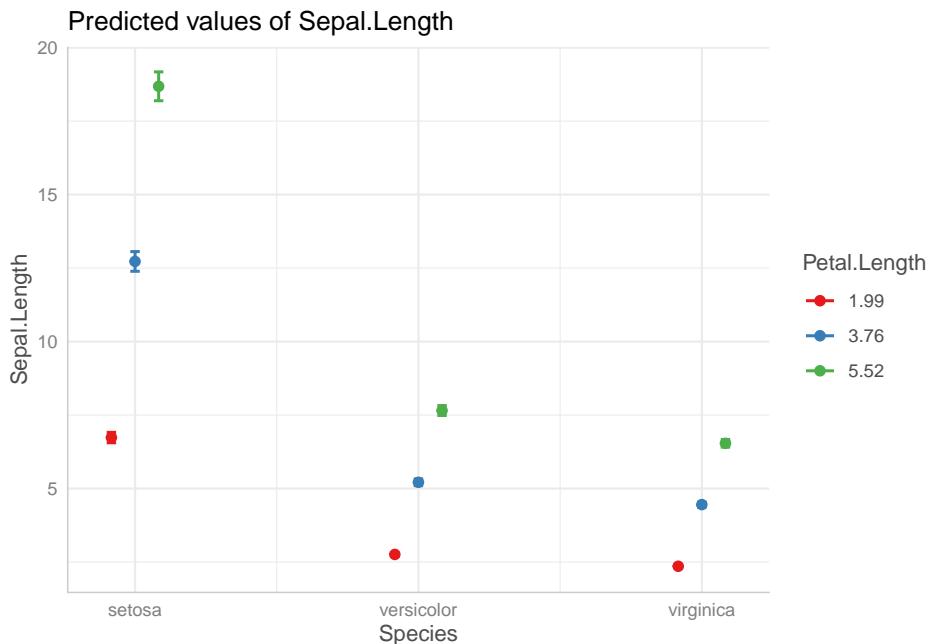
```
plot(ggeffect(m5, terms=c("Species", "Petal.Length")))
```



```
m6 <- lm(Sepal.Length ~ 0 + Petal.Length + Petal.Length:Species, data = iris)
plot(ggeffect(m6, terms=c("Petal.Length", "Species")))
```



```
plot(ggeffect(m6, terms=c("Species", "Petal.Length")))
```



Kaks statistikat: ajaloost ja tõenäosusest

Bayesiaanlik ja sageduslik statistika leiutati üksteise järel Pierre-Simon Laplace poolt, kes arendas välja kõigepealt bayesiaanliku statistika alused ning seejärel sagedusliku statistika omad (ca. 1774 - 1814). Sagedusliku statistika õitsengu põhjusteks 20. sajandil olid arvutuslik lihtsus ning tõenäosuse sagedusliku tõlgenduse sobivus 20 saj esimeses pooles käbinud teadusfilosoofiatega - eeskätt loogilise postivismiga. 1930-1980-ndatel valitseb akadeemiliste statistikute seas seisukoht, et Bayesiaanlik statistika on surnud ja maha maetud, ning selle arendamisega tegelesid vaid üksikud inimesed, kes sageli olid füüsikaliste teaduste taustaga (Jeffreys, Jaynes).

Alates 1960-e keskpaigast arendati bayesiaanlust USA sõjaväe egiidi all, kuna seal oli piisav juurdepääs arvutivõimsusele, kuid seda tehti paljuski salastatult. Bayesi meetoditega ei olnud võimalik korralikult tsiviilteadust teha enne 1990-ndaid aastaid, mil personaalarvutite levik algatas buumi nende meetodite arendamises. Praegu on maailmas bayesiaanlikku ja sageduslikku statistikat umbes pooleks (vähemalt uute meetodite arendustöö poole pealt). Eestis bayesiaanlik statistika 2017 aasta seisuga peaaegu, et puudub.

1930ndatel kodifitseeris Andrei Kolmogorov tõenäosusteooria aksioomid (3 aksioomi), mis ütlevald lühidalt, et tõenäosused jäädvad 0 ja 1 vahele ning, et

ükssteist välistavate ja hüpoteesiruumi ammendavate hüpoteeside tõenäosused summeeruvad ühele. Selgus, et Bayesi teoreem on lihtsa aritmeetika abil tuletatav Kolmogorovi aksioomidest. Tagantjärele saame öelda, et bayesiaanlik statistika on mitte ainult tõenäosusteooriga kooskõlas vaid ka, et Bayesi teoreem on parim võimalik viis sellist kooskõla saavutada (see on 1950ndate tarkus - Coxi teoreem). On ka teada, et kui tõenäosused on fikseeritud nulli ja ühega, siis taandub Bayesi teoreem klassikalisele lausearvutuslikule loogikale. See tähendab, et klassikaline loogika on bayesiaanluse erijuht. Seevastu sageduslik statistika püüab saavutada mõistlikke lahendusi arvutuslikult lihtsamate meetoditega, mille hinnaks on formaalse kooskõla puudumine tõenäosusteooriga. Seega kujutab sageduslik statistika endast kogumit *ad hoc* meetodeid, mis ei tähenda muidugi, et sellest kasu ei võiks olla. Küll aga tähendab see, et kuigi sageduslike mudeliteid on lihtsam arvutada, on neid raskem ehitada ja mõista ning, et sageduslike testide, milliseid on viimase saja aasta jooksul loodud 10 000 ringis, tulemusi on raskem tõlgendada.

Kahe statistika põhiline erinevus ei tulene tõenäosusteooria matemaatikast, vaid erinevatest tõenäosuse tõlgendusest.

Bayesi tõlgenduses on tõenäosus teadlase usu määr mingi hüpoteesi kehtimisse. Hüpotees võib näiteks olla, et järgmise juulikuu sademetel hulk Vilsandil jäääb vahemikku 22 kuni 34 mm. Kui Bayesi arvutus annab selle hüpoteesi tõenäosuseks 0,57, siis oleme me selle teadmise ajal nõus maksma mitte rohkem kui 57 senti kihlveo eest, mille alusel makstakse juhul, kui see hüpotees töeseks osutub, välja 1 EUR (ja me saame vähemalt 43 senti kasumit).

Sageduslikud teoreetikud usuvad, et selline tõenäosuse tõlgendus on ebateaduslik, kuna see on "subjektiivne". Nimelt on võimalik, et n teadlast arvutavad samade andmete, kuid erinevate taustateadmiste põhjal n erinevat korrektset tõenäosust. Veelgi hullem, kui nad lähtuvad väga erinevatest taustauskumustest, ei pruugi toimuda teadmiste konvergeerumist ka siis, kui nende arvutustesse andmeid järjest juurde tuua.

Sellise olukorra osaline analoogia poliitikas on elanikkond, kus pooled kannavad konservatiivseid väärtsusi (perekond, rahvusühtsus) ja teine pool liberaalseid (multikultuursus, üldnimlikud väärtsused). Seega priorid on erinevad. Senikaua kui mõlemad pooled saavad oma uudised samast allikast (sama tõepära), ei ole demokraatia siiski ohus. Aga kui pooled hakkavad erinevatest allikatest hankima erinevaid fakte, mis kummagi ideoloogiat kinnitavad, toimub arvamuste polariseerumine ning demokraatia sattub ohtu.

Seega, kui te usute, et teie taustateadmised ei tohi mõjutada järeldusi, mis te oma andmete põhjal teete, siis te ei ole bayesiaan. Siinkohal pakub alternatiivi tõenäosuse sageduslik tõlgendus. Sageduslik tõenäosus on defineeritud kui teatud tüüpi andmete esinemise pikajaline suhteline sagedus. Näiteks, kui me viskame münti palju kordi, siis peaks kullide (või kirjade) suhteline sagedus andma selle mündi tõenäosuse langeda kiri üleval. Selline tõenäosus on omistatav

ainult sellistele sündmustele, mille esinemisel on sagedus. Kuna teaduslikul hüpoteesil ei ole esinemise sagedust, ei ole sageduslikus statistikas võimalik rääkida ka hüpoteesi kehtimise tõenäosusest. See on tõsine probleem, kuna tüüpiline statistiline hüpotees ütleb: meie mõõtmiste keskväärtus jäab vahemikku a kuni b – ja me tahaksime sellele anda usaldusintervallid, mis ütleksid, millise tõenäosusega see hüpotees kehtib. Sageduslik lahendus on selle asemel, et rääkida meie hüpoteesi tõenäosusest meie andmete korral, rääkida andmete, mis sarnanevad meie andmetega, esinemise tõenäosusest null-hüpoteesi (mis ei ole meie hüpotees) kehtimise korral. Seega omistatakse sagedus ehk tõenäosus andmetele, mitte hüpoteesile. Samas bayesiaan, kelle tõenäosused kehtivad hüpoteesidele, räägib otse tõenäosuse, millega parameetri väärthus jäab a ja b vaheli, või ükskõik millisesse muusse teid huvitavasse vahemikku.

Poleemika: kumbki tõenäosus pole päris see, mida üldiselt arvatakse

Bayesi tõenäosus ei anna tegelikult seda tõenäosusnumbrit, mida me realselt peaksime kihlveokontoris kasutama. Ta annab numbri, milles me lähtuksime juhul, kui me usuksime, et selle numbri arvutamisel kasutatud statistilised mudelid kirjeldavad täpselt maailma. Paraku, kuna mudeldamine on oma olemuselt kompromiss mudeli lihtsuse ja ennustusvõime vahel, ei ole meil põhjust sellist asja uskuda. Seega ei peaks me bayesi tõenäosusi otse maailma üle kandma, vähemasti mitte automaatselt. Bayes ei ütle meile, mida me realselt usume. Ta ei ütle, mida me peaksime uskuma. Ta ütleb, mida me peaksime uskuma tingimuslikult.

Sageduslik tõenäosus on hoopis teine asi. Seda on võimalik vaadelda kahel viisil:

1. imaginaarsete andmete esinemissagedus nullhüpoteesi all;
2. reaalsete sündmuste esinemise sagedus.

Teise vaate kohaselt on sageduslik tõenäosus päriselt olemas. See on samasugune füüsikaline nähtus nagu näiteks auto kiirus, mõõdetuna liiklusmiilitsa poolt.

Kui kaks politseinikku mõõdavad sama auto kiirust ja 1. saab tulemuseks 81 km/h ning 2. saab 83 km/h, siis meie parim ennustus auto kiiruse kohta on 82 km/h. Kui aga 1. mõõtmistulemus on 80 km/h ja teine 120 km/h, siis meie parim hinnang ei ole 100 km/h. Enne sellise hinnangu andmist peame tegema lisatööd ja otsustama, kumb miilits oma mõõtmise kihva keeras. Ja me ei otsusta seda mitte oodatavast trahivist lähtuvalt, vaid neutraalseid objektiivseid asjaolusid vaagides. Seda sellepärast, et autol on päriselt kiirus olemas ja meil on hea põhjus, miks me tahame seda piisava täpsusega teada. Sagedusliku statistiku mõõteriist on statistiline mudel ja mõõtmistulemus on tõenäosus, mis jäab 0 ja 1 vaheli.

Õpikunäidetes on sündmusteks, mille esinemise sagedust töenäosuse abil mõõdetakse, enamasti täringuvisked, ehk katsesüsteemi reaalne füüsikaline funktsioneerimine. Pane tähele, et need on inimtekkelised sündmused (loodus – ega jumal – ei viska täringuid). Teaduses on sündmused, millele töenäosusi omistatakse, samuti inimtekkelised: selleks sündmuseks on teadlase otsus H_0 ümber lükkamise kohta, mille tegemisel ta lähtub p (või q) väärtestest ja usaldusnivoost. Siin vastab auto kiirusele 1. tüüpi vigade tegemise sagedus. See sagedus on inimtekkeline, aga selles hoolimata päriselt olemas ja objektiivselt mõõdetav. Kui 2 teadlast mõõdavad seda paraleelselt ja saavad piisavalt erineva tulemuse (näiteks väga erineva FDR-i), võib olla kindel, et vähemalt üks neist eksib, ning peaks olema võimalik ausalt otsustada, kumb.

Võrdlev näide: kahe gruvi võrdlus

Järgnevalt toome näite, kuidas bayesiaan ja sageduslik statistik lahendavad sama ülesande. Meil on 2 gruupi, katse ja kontroll, millega kummagi 30 mõõtmist ja me soovime teada, kui palju katsetingimus mõjutab mõõtmistulemust. Meie andmed on normaaljaotusega ja andmepunktid, mida me analüüsime, on efektisuurused (katse1 - kontroll1 = ES1 jne).

Bayesiaan

Statistiline küsimus on Bayesiaanil ja sageduslikul statistikul sama: kas ja kui palju erinevad kahe gruvi keskväärtused? Bayesiaan alustab sellest, et ehitab kaks mudelite: andmete töepäramudel ja taustateadmiste mudel ehk prior.

Kui andmed on normaaljaotusega, siis on ka töepäramudel normaaljaotus. Alustame sellest, et fitime oma valimiandmed (üksikud efekti suurused) normaaljaotuse mudelisse.

See ei ole veel töepäramudel, sest me tahame hinnangut ES **keskväärtuse** kõige töenäolisemale väärtsusele, ja lisaks veel hinnangut ebakinlusele selle punkt-hinnangu ümber (usalduspiire). Seega tuleb eelmine jaotus kitsamaks tõmmata, et ta kajastaks meie teadmisi ES-ide keskväärtuste, mitte individuaalsete ES-de, kohta. Uue jaotusmudeli sd = eelmise jaotuse sd/sqrt(30).

Täpsemalt, selle joonise põhjal võib arvutada, milline on meie valimi keskväärtuse kohtamise töenäosus igal võimalikul töelisel ES-i väärtsel. Kõige töenäolisemad on andmed siis, kui tegelik ES = andmete keskväärtusega (seda kohta näitab must joon). Kui me jagame musta joone pikkuse punase kurvi all läbi katkendjoone pikkusega sama kurvi all, saame teada, mitu korda on meie andmed töenäolisemad siis, kui tegelik ES = mean(valimi ES), vörreldes olukorraga, kus tegelik ES = 0. Loomulikult võime sama näitaja arvutada ükskõik millise hüpotheseside paari kohta (näiteks, andmed on miljon korda töenäolisemad hüpoteesi

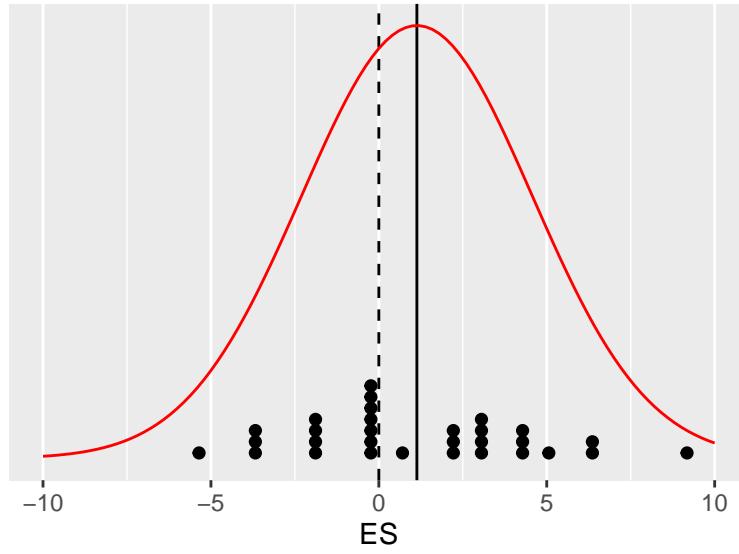


Figure B.1: Paariviisiline katse - kontroll disain. Katset on korratud 30 korda. X-teljel on efektisuurused (ES). 30 üksikut efektisuurust on näidatud punktidena. Must joon näitab keskmist efektisuurust. Andmed on mudeldatud normaaljaotusena.

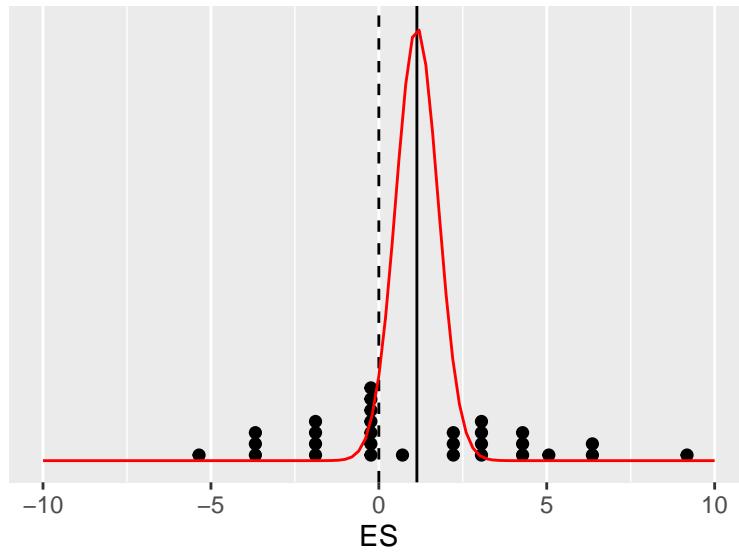


Figure B.2: See jaotus iseloomustab keskmise ES paiknemist puhtalt meie andmete põhjal.

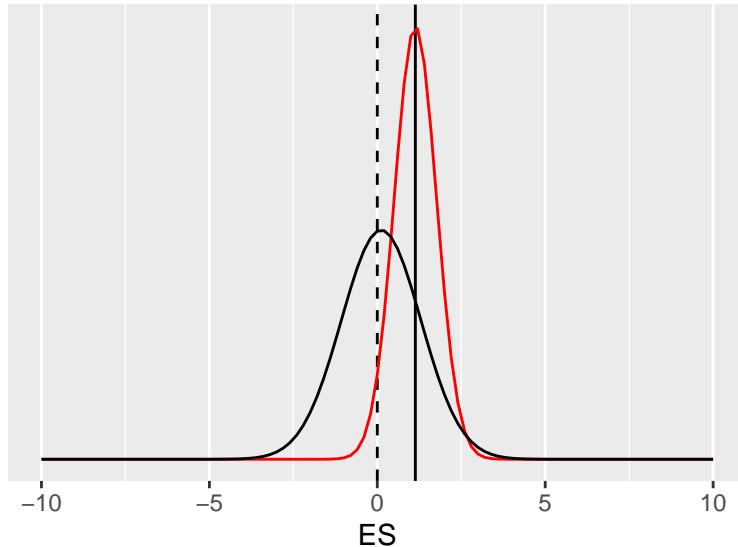


Figure B.3: Taustateadmiste mudel ehk prior on normaaljaotus (must joon), mille ülesanne on veidi vähendada ekstreemsete valimite kahjulikku mõju.

$ES = 0.02$ all kui hüpoteesi $ES = -1$ all; mis aga ei tähenda, et andmed oleksid väga töenäolised kummagi võrreldud hüpoteesi all).

Aga see ei ole veel Bayes. Lisame andmemudelile taustateadmiste mudeli. Sellega tühistame me väga olulise eelduse, mis ripub veskikivina sagedusliku statistika kaelas. Nimelt, et valimi andmed peavad olema esinduslikud populatsiooni suhtes. Me võime olla üsna kindlad, et väikeste valimite korral see eeldus ei kehti ja sellega seoses ei tööta ka sageduslik statistika viisil, milleks R.A. Fisher selle kunagi lõi. Taustateadmiste mudeli peamine, kuigi mitte ainus, roll on mõjutada meie hinnangut õiges suunas vähendades halbade andmete võimet meile kahju teha. Kui sul on väike valim, siis sinu andmed vajavad sellist kantseldamist.

Olgu meie taustateadmise mudel normaaljaotus keskväärtusega 0 ja standardhälbega 1.

Taustateadmiste mudel on sageli normaaljaotus. Kui meil on palju taustateadmisi, siis on see jaotus kõrge ja kitsas, kui meil on vähe taustateadmisi, siis on see madal ja lai.

Mida teha, kui sa ei taha, et taustateadmiste mudel sinu posteeriori kuju mõjutab? Sellisel juhul kasutatakse nõrgalt informatiivseid prioreid, mis tähendab, et priori jaotus on palju laiem kui tõepäramudeli laius. Miks mitte kasutada mitte-informatiivseid tasaseid prioreid? Põhjused on arvutuslikud, seega tehnilist laadi.

Igal juhul järgmise sammuna korrutab bayesiaan selle jaotuse andmejaotusega,

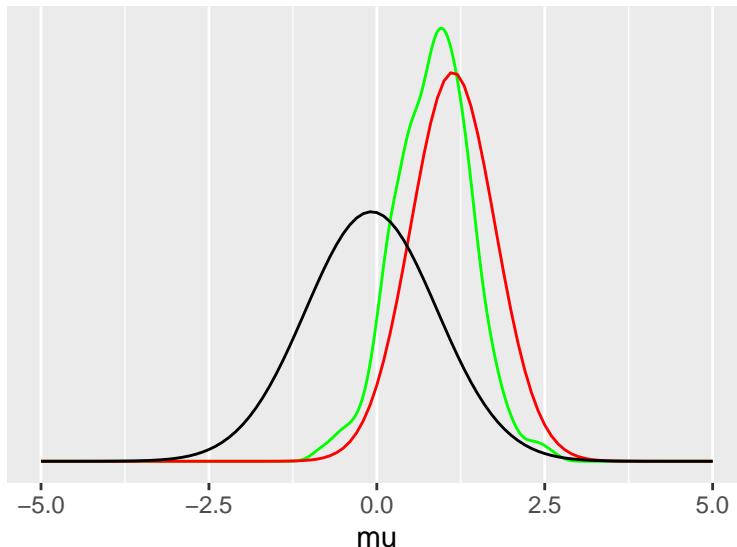


Figure B.4: Triplot. Bayesi väljund on posterioorne tõenäosusjaotus (roheline). Nagu näha, ei ole selle jaotuse tipp täpselt samas kohas kui andmejaotuse tipp ehk keskväärtus. Prior tömbab seda veidi nulli suunas. Lisaks on posteerior veidi kitsam kui andmemudel, mis tähendab, et hinnang ES-le tuleb väiksema ebakindluse määraga.

saades tulemuseks kolmenda normaaljaotuse, mille ta seejärel normaliseerib nii, et jaotuse alune pindala = 1. See kolmas jaotus on posterioorne tõenäosusjaotus, mis sisaldab kogu infot, millega saab arvutada kõige tõenäolisema katseefekti suuruse koos ebakindluse määraga selle ümber (mida rohkem andmeid, seda väiksem ebakindlus) ja tõenäosused, et tegelik katseefekt jääb ükskõik milllisesse meid huvitavasse vahemikku.

Nüüd ei ole siis muud kui bayesi mudel läbi arvutada.

```
dfa <- data.frame(a)
m99 <- map2stan(
  alist(
    a ~ dnorm(mean = mu, sd = sigma),
    mu ~ dnorm(0, 1),
    sigma ~ dcauchy(0, 1)),
  data = dfa)
```

Posteerior sisaldab endas kogu infot, mis meil ES-i tõelise väärtsuse kohta on. Siit saame arvutada:

1. parima hinnangu ES-i punktväärtusele,
2. usaldusintervalli, ehk millisest ES-ide vahemikust loodame leida tõelise

- ES-i näit 90% tõenäosusega,
3. iga mõeldava ES-i väärustete vahemiku kohta tõenäosuse, millega tõeline ES jääb sellesse vahemikku.
 4. saame ES-i põhjal arvutada mõne muu statistiku, näiteks $ES1 = \log(ES)$, kasutades selleks ES-i posterioorset jaotust. Sel viisil kanname oma ES-i hinnangus peituva ebakindluse üle ES1-le, millele saame samuti rakendada punkte 1-3 (sest ES1 on posterioorne jaotus).
 5. uute andmete lisandumisel saame kasutada ES-i posteeriorit uue priorina ja arvutada uue täiendatud posteeriori. Põhimõtteliselt võime seda teha pärast iga üksiku andmepunkti lisandumist. See avab ka head võimalused metaanalüüsiks.
 6. lisaks saame oma algsest mudelist ka posteeriori andmepunkti tasemel varieeruvusele (pole näidatud). Seda kasutame uute andmete simuleerimiseks (meie näites üksikud ES-d).

Sageduslik statistik

Sageduslik lähenemine sisaldab ainult ühte mudelit, mida võrreldakse valimi andmetega. Sageduslik statistik alustab selles lihtsas näites täpselt samamoodi nagu bayesiaan, tekitudes eelmisega identse andmemudeli, mis on keskendatud valimi keskväärtusele B.2. Seejärel nihutab ta oma andmemudelit niipalju, et normaaljaotuse tipp ei ole enam valimi keskväärtuse kohal vaid hoopis 0-efekti kohal. Jaotuse laius nihutamisel ei muutu.

Seda nullile tsentreeritud mudelit kutsutakse null-hüpoteesiks (H_0). Nüüd võrdleb ta oma valimi keskväärtust (must joon) H_0 jaotusega. Kui valimi keskväärtuse kohal on H_0 jaotus kõrge, siis on andmete tõenäosus H_0 kehitmise korral suur. Ja vastupidi, kui valimi keskväärtuse kohal on H_0 madal, siis on andmete esinemise tõenäosus H_0 all madal. Seda tõenäosust kutsutakse p vääruseks. Mida väiksem on p , seda vähem tõenäolised on teie andmed juhul, kui H_0 on tõene ja katseefekt võrdub nulliga. P on defineeritud kui “teie andmete või 0-st veel kaugemal asuvate andmete esinemise pikaajaline suhteline sagedus tingimusel, et H_0 kehtib”.

Tulemuste tõlgendamine

Kui sageduslik statistik kirjutab, et tema “efekti suurus on statistiliselt oluline 0.05 olulisusnivool”, siis ta ütleb sellega, et tema poolt arvutatud $p < 0.05$. Selle väite korrektna tõlgendus on, et juhul kui statistik pika aja jooksul võtab omaks “statistikiliselt olulistena” kõik tulemused, millega kaasnev $p < 0.05$ ja lükkab tagasi kõik tulemused, mille $p > 0.05$, siis sooritab ta 5% sagedusega 1. tüüpivigu. See tähendab, et igast sajast tõesest H_0 -st, mida ta testib, võtab ta

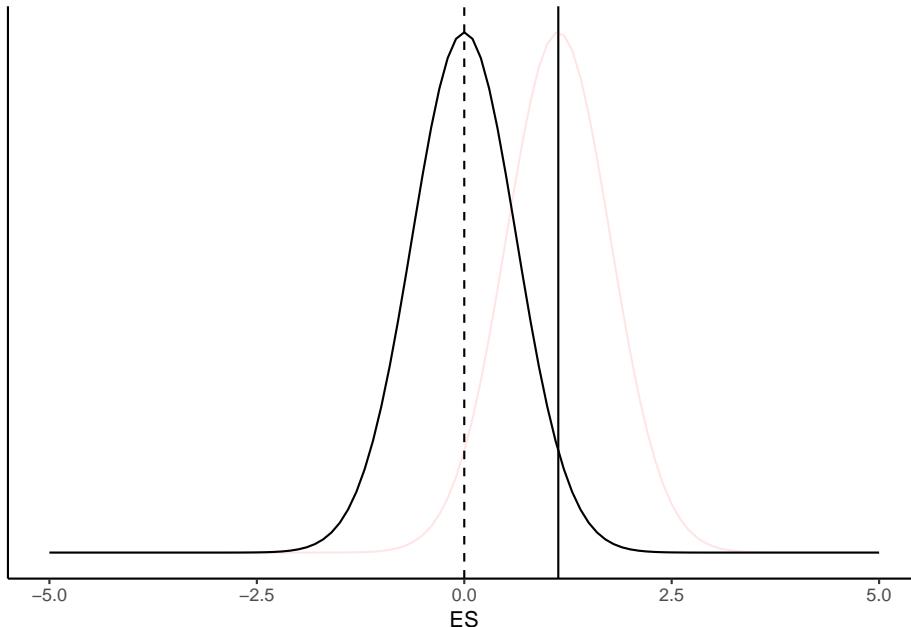


Figure B.5: Nullhüpotees (must kõver) ja tõepärafunktsioon (punane kõver).

keskelt läbi 5 vastu, kui statistiliselt olulised. Sageduslik statistika on parim viis 1. tüüpi vigade sageuse fikseerimiseks.

Paraku ei tea me ühegi üksiku testi kohta ette, kas see testib kehtivat või mittekehtivat H_0 -i, mis teeb raskeks katseseeriate ühekaupa tõlgendamise. Tuletame meelete, et sageuslikus statistikas ei saa rääkida H_0 kehtimise tõenäosusest vaid peab rääkima andmete tõenäosusest (ehk andmete esinemise sageusest) tingimusel, et H_0 kehtib.

Kas ühte p väärust saab tõlgendada kui hinnangut tõendusmaterjali hulgale, mida teie valim pakub H_0 vastu? Selle üle on vaieldud juba üle 80 aasta, kuid tundub, et ainus viis seda kas või umbkaudu teha on bayesiaanlik. Igal juhul, p väärust, mis on defineeritud pikaajalise sageusena, on raske rakendada üksiksündmusele. Bayesiaanliku p vääruste tõlgendamiskalkulaatori leiate aadressilt <http://www.graphpad.com/quickcalcs/interpretPValue1/>.

Kujutle mass spektroskoopia katset, kus mõõdame 2000 valgu taseid katse-kontroll skeemis ja katset korratakse n korda. Sageduslik statistik kasutab adjusteeritud p väärusti või q väärusti, et tömmata piir, milles ühele poole jäavat statistiliselt olulised ES-d ja teisele poole mitteolulised null-efektid. Edasi tõlgendab ta mitteolulisi efekte kui ebaolulisi ja diskuteerib vaid “olulisi” efekte. Paraku, p vääruste arvutamine ja adjusteerimine saab toimuda mitmel erineval moel

ja usalduspiiri panekule just 95-le protsendile, mitte näiteks 89% või 99.2%-le, pole ühtegi ratsionaalset põhjendust. Seega tömbab ta sisuliselt juhuslikus kohas joone läbi efektide, misjärel ignoreerib kõiki sellest joonest valele poole jäänud efekte. Meetod, mis väga hästi töötab pikaajalisest kvaliteedikontrollis, ei ole kahjuks kuigi mõistlik katse tulemuste ükshaaval tõlgendamises. Mis juhtub, kui oleme kavalad ja proovime mitmeid erinevaid p väärustega töötamise meetodeid, et valida välja see usalduspiir, millest õigele poole jääävaid andmeid on teaduslikult kõige parem tõlgendada? Ehkki ükshaaval võisisid kõik meie poolt läbi arvutatud meetodid olla lubatud (ja isegi võrdsest head), ei fikseeri p nüüd enam 1. tüüpiligi vigade sagedust. See tähendab, et p on kaotanud definitsioonijärgse tähenduse ja te oleksite võinud olulisuspiiri sama hästi tõmmata tunde järgi.

Tüüpiline tulemuse kirjeldus artiklis:

1. sageduslik: *the effect is statistically significant ($p < 0.01$)*.
2. bayesiaanlik: *the most likely effect size is xxx (90% CI = xxx-low, xxx-high) and the probability that the true effect is < 0 is xxx percent.*

90% CI — *credible interval* — tähendab, et me oleme 90% kindlad, et tegelik efekti suurus asub meie poolt antud vahemikus.

Kahe paradigma erinevused

1. sageduslikus statistikas võrdub punkt-hinnang tegelikule efekti suurusele valimi keskmise ES-ga. Bayesi statistikas see sageli nii ei ole, sest taustateadmiste mudel mõjutab seda hinnangut. Paljud mudelid püüavad ekstreemseid valimeid taustateadmiste abil veidi mõistlikus suunas niutada, niiviisi vähendades ülepaisutatud efektide avaldamise ohtu.
2. sageduslik statistika töötab tänu sellele, et uurija võtab vastu pluss-miinus otsuseid: iga $H \sim 0$ kas lükatakse ümber või jäetakse kehtima. Seevastu bayesiaan mõtleb halli varjundites: sissetulevad andmed kas suurendavad või vähendavad hüpoteeside tõenäosusti (mis jäavat aga alati > 0 ja < 1).
3. p vääritud kontrollivad 1. tüüpiligi vigade sagedust ainult siis, kui katse disaini ja hilisema tulemuste analüüs detailid on enne katse sooritamist fikseeritud (või eelnevalt on täpselt paika pandud lubatud variatsioonid katse- ja analüüs protokollis). Eelkõige tähendab see, et valimi suurus ja kasutatavad statistilised testid peavad olema eelnevalt fikseeritud. Tüüpiliselt saame p vääruse arvutada vaid üks kord ja kui $p = 0.051$, siis oleme sunnitud $H \sim 0$ paika jätma ning efekti deklareerimisest loobuma. Me ei saa lihtsalt katset juurde teha, et vaadata, mis juhtub. Bayesiaan seevastu võib oma posterioorse tõenäosuse arvutada kasvõi pärast iga

katsepunkti kogumist ning katse peatada kohe (või alles siis), kui ta leiab, et tema posterioorne jaotus on piisavalt kitsas, et teaduslikku huvi pakkuda.

4. Sageduslikus statistikas sõltub tulemus sellest, kas hüpotees, mida testitakse oli defineeritud enne andmete kogumist või mitte. Selle pärast tuleks seal testitavad hüpoteesid eel-registreerida. Bayesi statistikas pole aga vahet, kas hüpotees on formulieeritud enne või pärast andmete nägemist – seal loevad töepärafunktiooni loomisel ainult olemasolevad andmed (mitte andmed, mis oleksid võinud olla aga ei ole).
5. sagedusliku statistika pluss-miinus iseloom tingib selle, et kui tegelik efekti suurus on liiga väike, et sattuda õigele poole olulisusnivood, siis annavad statistiliselt olulisi tulemusi ülepaisutatud efektid, mida tekib tänu valimiveale. Nii saab süsteematiiliselt kallutatud teaduse. Bayesi statistikas seda probleemi ei esine, kuna otsused ei ole pluss-miinus tüüpi.
6. bayesi statistika ei fikseeri 1. tüüpi vigade sagedust. See-eest võitleb see nn valehäirete vastu, milleks kaasajal kasutatakse enim mitmetasemelisi *shrinkage* mudeliteid. See on bayesi vaste sageduslikus statistikas kasutatavatele mitmese testimise korrektsioonidele. Kui sageduslik statistik võitleb valehäiretega p väärtsi adjusteerides ja selle läbi olulisusnivood nihutades, siis bayesiaan kasutab *shrinkage* mudelit, et parandada hinnanguid üksikute efektide keskväärtustele ja nende sd-le, kasutades paindlikult kogu andmestikus leiduvat infot.

Sageduslik ja teaduslik hüpoteesitestimine

Teaduslik lähenemine töendusmaterjalile on sarnane kohtuliku uurimisega: me kogume töendusmaterjali senikaua, kuni oleme veendumud, et saame selle põhjal eelistada ühte hüpoteesi kõikide teiste arvelt. Seega, kui faktid meile piisavat surve avaldavad, võtame vastu pluss-miinus otsuse, et kohtualune süüdi või teaduslik hüpotees õigeks mõista. See otsus on meie tegevuse eesmärk ja meie tegevus oli suunatud selle eesmärgi täitmisele.

Sageduslik statistika võtab samuti vastu dihhotoomseid otsuseid, aga hoopis teisel moel. Seal ei ole otsus eesmärk, vaid vahend. Kui me lükkame tagasi teatud null hüpoteesid, aga mitte teised, saame me sellisel viisil fikseerida pikaajalise 1. tüüpi vigade sageduse. Me võtame vastu otsuseid, eesmärgiga tagada katsesüsteemi pikaajaline kvaliteet. Need otsused ei eelda, et me usuksime, et mõni konkreetne null hüpotees on tõene või väär ning matemaatilised protseduurid, mille alusel me neid otsuseid langetame, ei püüa määräta individuaalse null hüpoteesi tõelähedust või tõenäosust, et see H_0 ei kehti. Seega ei tähenda fakt, et me lükkasime ümber konkreetse nullhüpoteesi seda, et me usume, et see null hüpotees on väär.

H_0 -i ümber lükkamisel põhineva statistikaga kaasnevad järelmid, millest ehk olulisim on vajadus fikseerida andmete kogumise ja analüüsmeetodid enne, kui andmed on kogutud. See on nii tehnilikel põhjustel, mis on seotud puhtalt meie

sooviga fikseerida 1. tüüpi vigade sagedus. Sellise veasageduste fikseerimise hind on, et andmeanalüüs valikud ei saa sõltuda tegelikest andmetest (nende kvaliteedist, jaotusest jms). Siinkohal tuleb eraldi röhutada, et tegemist ei ole üldise teadusliku meetodi omadusega. Teaduses (ja bayesi statistikas) on mitte ainult täiesti normaalne vaid lausa vajalik vaadata andmeid kriitilise pilguga ja kujundada oma analüüs vastavalt andmete kvaliteedile. Ning kui andmed ei paku piisavalt töendusmaterjali, et me saaksime otsustada oma hüpoteesi kasuks või kahjuks, siis on igati mõistlik andmeid juurde korjata senikaua, kuni oleme veendunud ühte või teistpidi.

Oluline erinevus sagedusliku ja bayesi statistika vahel on, et kui sageduslik meetod fikseerib pikaajalise veasageduse aga ei arvuta üksikute hüpoteeside tõenäosust, siis bayesi meetod vastupidi arvutab üksikute hüpoteeside tõenäosused, aga ei fikseeri pikaajalisi veasagedusi. Kui meid ikkagi huvitavad veasagedused ja statistiline võimsus, saab neid ka bayesiaanlikult leida, arvutades oma mudeleid simuleeritud andmetega.

Statistiline ennustus kui mitmetasandiline protsess

Me võime vaadelda ennustavat statistikat mitmetasemelise protsessina, kus alusel tasemel on punkthinnang parameetri väärtsuse, selle pealoleval tasemel on hinnang ebakindlusele selle punkthinnangu ümber, ning 3. tasemel on omakorda hinnang ebakindlusele 2. taseme hinnangu ümber. Ja nii edasi lõpmatusse. Bayes erineb klassikalisest statistikast selle poolest, et kui Bayes ehitab 2. taseme hinnangu töepära ja priori põhjal, siis klassikaline statistika kasutab selleks pelgalt töepära (konverteerituna null hüpoteesiks). See on tähtis, kuna töepära modelleerib ainult seda osa juhuslikust varieeruvusest punkthinnangu ümber, mida kutsutakse valimiveaks. Prior on võimeline arvesse võtma ka andmete kallutatust.

Kuna klassikalises statistikas ei ole formaalset priori mudelit, ei hindata klassikalised usaldusintervallid (2. tase) töelist ebakindlust punktväärtuse ümber. Seda teevald bayesiaanlikud krediibiilsusintervallid, aga ainult siis, kui priorite koostamisse on tõsiselt suhtutud.

I Punkthinnang – enamasti aritmeetiline keskmene — modelleerib andmejaotuse tüüpilist elementti. Eeldus: me teame, milline on andmete jaotus.

II töepärafunktsioon hindab ebakindlust punkthinnangu ümber. Modelleerib valimiviga, mis on seda suurem, mida vähem on teil andmeid. Eeldus 1: andmed on esinduslikud (andmejaotus = populatsiooni jaotus) Eeldus 2: mudel kirjeldab andmeid genereerivat mehhaniismi (siit tulevad sageli lisaeeldused, nagu populatsiooni normaaljaotus, lineaarsus, sõltumatud sündmused valimi koostamisel, vigade sõltumatus, homoskedastilus jms)

III prior kohendab töepärafunktsiooni hinnangut Modelleerib (1) andmete esinduslikkust, mis on seda väiksem, mida väiksem on valim, ja (2) süstemaatilist viga. Eeldus: meil on oma andmetest sõltumatuid teadmisi populatsiooni jaotuse kohta

Kui valim on piisavalt suur, siis võime olla piisavalt kindlad, et andmed on esinduslikud ning klassikalise statistika hinnangud ebakindlusele punktväärtuse ümber muutuvad selle võrra usutavamaks.

Samas, sedamööda kui valimi suurus kasvab, muutub töepärafunktsioon üha kitsamaks, mis tõstab omakorda tõenäosust, et tegelik parameetri väärthus jäab töepärafunktsiooni kõrgema osa alt välja, tingituna süstemaatilisest veast, mille suurus ei sõltu valimi suurusest. Seega töötab klassikaline statistika parimini keskmiselt suurte valimite (ja keskmiselt suure andmete varieeruvuse) korral.

B.0.1 Ajaloolist juttu: normaaljaotus, Bayes ja sageduslik statistika

(Anders Hald; A History of Parametric Statistical Inference from Bernoulli to Fisher, 1713-1935, Springer 2000)

Laplace sõnastas 1774. aastal statistiku tööpöllu järgmiselt: kirjeldamaks andmeid (vigade jaotust) tõenäosusfunktsioonina leia matemaatiline mudel, millel oleks lõplik arv parameetreid. Seejärel leia algoritm, mis minimeeriks vead meie hinnagutele nende parameetrite väärustele kohta. Seega oli eesmärk konverteerida andmete (mõõtmisvad) jaotus posterioorseks tõenäosusjaotuseks, mille pealt saaks omakorda arvutada usaldusintervallid meie hinnagu täpsusele.

Laplace, kes tegeles palju astronoomiliste mõõtmiste analüüsiga, lootis näidata, et mõõtmisandmete aritmeetiline keskmine on parim viis arvutada sellise posterioorse jaotuse kõige tõenäolisemat väärust (lokatsiooniparameetrit). Parim viis selles mõttes, et teoreetiliselt parima lokalisatsiooniparameetri ümber on võimalik arvutada kitsaimad veapiirid. Aritmeetilise keskmise selle pärast, et selle kasutamine oli laialt levinud, tundus intuitiivselt mõistlik ning oli arvutuslikult lihtne. Laplace probleemi lahendas Gauss ca. 1809, võttes kasutusele nii uue tõenäosusjaotuse – normaaljaotuse – kui ka uue lokatsiooniparameetri arvutusmeetodi – vähimruutude meetodi. Tema küsitud küsimus oli: millist jaotust ja hindamismeetodit oleks vaja kasutada, et lokatsiooniparameeter tuleks just aritmeetiline keskmine? Rõhutades normaaljaotuse tähtsust, leidis Laplace 1812. aastal, et köikidest sümmetrilistest andmejaotustest viib ainult normaaljaotus olukorrani, kus aritmeetiline keskmine kattub posterioorse jaotuse tipuga.

Kuna Laplace ei teadnud midagi tegelike veajaotuste kohta astronoomilistel mõõtmistel, oli tal raskusi andmejaotuse spetsifitseerimisega, mille pealt Bayesi teoreemiga posterioorne hinnanguvad jaotus arvutada. Mõõtmisvigu tavatseti mudeldada nelinurksete, kolmnurksete või kvadraat-jaotustega ja polnud ühtki teaduslikku põhjust eelistada üht jaotust teistele. See oli üks põhjuseid, miks Laplace hakkas arendama sageduslikku statistikat ja kasutas alates 1811 üha

vähem Bayesi teoreemi. (Alles 1818 näitas Bessel empiiriliselt, et astronoomilised mõõtmised on normaaljaotusega.) Teine põhjus hüljata Bayesi statistika oli seotud tehniliste raskustega priorite mudeldamisel, mistõttu Laplace oli sunnitud kasutama tasaseid prioreid, mis omistasid igale sündmusele/hüpoteesile võrdse eeltõenäosuse – ja oma ilmses absurduses tegid elu lihtsaks tema kriitikutele.

Sageduslik statistika kasutab oma alusena keskset piirteoreemi (Laplace 1810, 1812), mille kohaselt on paljude andmevalimite *aritmeetilised keskmised* normaaljaotusega, ja seda hoolimata andmete tegelikust jaotusest (eeldusel, et valimid on piisavalt suured; vt 6. ptk). Seega, senikaua kuni me ei modelleeri mitte ühe valimi empiirilist andmete jaotust (mida vajab Bayesi teoreem), vaid hoopis paljude virtuaalsete valimite pealt arvutatud *keskväärtuste jaotust*, ei pea me teadma, milline on andmete tegelik jaotus. Sellelt pinnalt ongi välja töötatud nullhüpoteesi testimine, millel põhineb suur osa 20. sajandi statistikast. 1908 näitas Edgeworth, et suurte valimite ja mitteinformatiivsete priorite korral annavad mõlemad meetodid (Bayes ja sageduslik) sama hinnangu parameetrväärtusele ja numbriliselt sama usaldusintervalli.

Sagedusliku statistika põhiprintsiibid: Fisher 1922.

Statistilisete meetodite eesmärk on andmete redutseerimine. Selleks on vaja vaadelda andmeid juhuvalimina hüpoteetilisest lõpmata suurest populatsioonist, mille jaotust saab kirjeldada suhteliselt väheste parameetritega mudeli abil. (Valimit iseloomustab “statistik”, populatsiooni aga “parameeter”)

Statistik puutub kokku kolme sorti probleemidega:

1. Spetsifikatsiooni probleemid kerkivad esile seoses populatsiooni jaotusmudeli spetsifitseerimisega.
2. Estimatsiooni probleemid kerkivad esile seoses algoritmidega, mille abil määratatakse hüpoteetilise populatsionimudeli parameetrite väärtsused.
3. Jaotuse probleemid kerkivad üles seoses valimite põhjal arvutatud statistikute jaotuste matemaatilise kirjeldamisega.

Heal estimatsioonil on omakorda kolm kriteeriumit:

1. Konsistentsus — statistik on kosnsistentne siis, kui arvutatuna lõpmata suurest valimist, mis võrdub populatsiooniga, tuleb selle statistiku väärtsus täpselt õige.
2. Efektiivsus — statistiku efektiivsus on suhe (protsent) selle sisemisest täpsusest võrrelduna teoreetiliselt efektiivseima statistikuga. Efektiivsus väljendab, kui suurt osa kogu kättesaadavast informatsioonist meie statistik kasutab. Efektiivsuse kriteerium: statistik, arvutatuna suurest valimist, annab vähima võimaliku standardhälbgaga normaaljaotuse.
3. Piisavus (sufficiency) — kriteerium: statistik on piisav kui ükski teine statistik, mida saab samast valimist arvutada, ei lisa informatsiooni hinnatava parameetri väärtsuse kohta.

Fisherit poolt juurutatud sageduslik terminoloogia: parameeter, statistik, tõepära, dispersioon (variance; ANOVA), konsistentsus, efektiivsus, piisavus, infomatsioon, null hüpotees, statistiline olulisus, p väärthus, olulisuse test, protsendipunkt, randomiseerimine, interaktsioon, faktoriaalne disain.

Tänu kesksele piirteoreemile on sageduslik statistika arvutuslikult palju lihtsam kui Bayesi statistika (arvutused saab teha paberil ja pliiatsiga), aga kuna me ei saa enam rääkida tegelike empiiriliste andmete jaotusest, vaid peame selle asemel kasutama lõpmata hulka virtuaalseid valimeid, ei saa me enam Bayesi teoreemi abil tõenäosusi pöörata (konverteerida meie andmete tõenäosus parameetriväärtuse x kehtimise korral, parameetriväärtuse x kehtimise tõenäosuseks meie andmete korral). Veelgi enam, me ei saa isegi rääkida meie andmete tõenäosusest parameetriväärtuse x kehtimise korral, vaid oleme sunnitud oma keelt ja meelt murdes rääkima "meie andmete või neist ekstreemsemate andmete pikajalisest suhtelisest sagedusest nullhüpoteesi (aga kahjuks mitte ühegi teise parameetriväärtuse) kehtimise korral" (Fisher ca. 1920). Nagu näha, on siin arvutuslikul lihtsuse sel kõrge kontseptuaalne hind.

Achen, Christopher H, and Larry M Bartels. 2016. *Democracy for Realists: Why Elections Do Not Produce Responsive Government*. Princeton University Press.