Paikkatietoaineistosta puuttuvan datan mallintaminen Markovin satunnaiskentillä

Esa Junttila

Helsinki 11.12.2006

Pro gradu -tutkielma

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta/Osasto — Fakultet/Sektion — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Esa Junttila

Työn nimi — Arbetets titel — Title
Paikkatietoaineistosta puuttuvan datan mallintaminen Markovin satunnaiskentillä

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu -tutkielma	11.12.2006	84 sivua + 14 liitesivua

Tiivistelmä — Referat — Abstract

Jos kerättyyn dataan liittyy sijaintitieto, niin datasta voidaan muodostaa paikkatietoaineisto. Kerätyistä paikkatietoaineistoista puuttuu usein dataa tai ne ovat muuten epätarkkoja. Aineistojen epävarmuus on usein jätetty huomiotta laskennallisten syiden vuoksi, jolloin jatkoanalyysinkin tulokset ovat epävarmoja. Puuttuvaa dataa voidaan mallintaa, jos datassa esiintyy riippuvuuksia: esimerkiksi lähialueet ovat yleensä varsin samankaltaisia.

Tässä tutkielmassa pyritään näyttämään, että puuttuvan datan mallintaminen on laskennallisesti mahdollista myös suurilla paikkatietoaineistoilla. Mallintamisessa käytetään bayesiläistä todennäköisyyslaskentaa ja Markovin satunnaiskenttiä, erityisesti Ising-mallia. Malleja käsitellään Markovin ketju Monte Carlo -menetelmällä, jotta tulosten tuottaminen olisi laskennallisesti tehokasta.

Tutkielmassa esiteltyjä menetelmiä käytetään kahteen todelliseen paikkatietoaineistoon: suomen murresanojen aineistoon ja Suomen lintujen pesimäaineistoon. Johtopäätös on, että puuttuvaa dataa voidaan mallintaa uskottavasti jo hyvin yksinkertaisilla malleilla. Lisäksi puuttuvan datan mallintaminen on laskennallisesti mahdollista, vaikkakin aikaa vievää. Tämä rohkaisee käyttämään esiteltyjä menetelmiä muihinkin puutteellisiin paikkatietoaineistoihin, jotta aineiston ja jatkoanalyysin tulosten epävarmuutta saataisiin vähennettyä.

ACM Computing Classification System (CCS):
I.6 [Simulation and Modeling],
G.3 [Probability and Statistics]

Avainsanat — Nyckelord — Keywords
Markovin satunnaiskentät, bayesiläiset hierarkkiset mallit, Markovin ketju Monte Carlo

Säilytyspaikka — Förvaringsställe — Where deposited
Kumpulan tiedekirjasto, sarjanumero C-

Muita tietoja — övriga uppgifter — Additional information

Sisältö

Liitteet

1 Johdanto

Monien kerättyjen aineistojen havaintoihin voidaan liittää sijaintitieto. Esimerkiksi murresanan käyttö voidaan paikallistaa johonkin kuntaan. Tällaiset aineistot ovat kuitenkin usein epätarkkoja tai muulla tavalla puutteellisia. Data-analyysissä on kuitenkin yleistä jättää suuressa aineistossa esiintyvä epävarmuus huomiotta ja käsitellä aineistoa sellaisenaan – laskennallisista syistä. Puuttuvan datan erityinen käsittely on haasteellista, mutta silti tarpeellista, jotta aineiston laatua voitaisiin parantaa. Jos tässä onnistutaan, niin aineiston mahdollinen jatkokäsittely tuottaa tarkempia tuloksia.

Sijaintitietoa sisältävän aineiston havainnot voivat riippua toisistaan: sijainniltaan toisiaan lähellä olevat havainnot ovat usein samankaltaisia. Aineiston puutteita voidaan korjata havaintojen välillä mahdollisesti esiintyvien riippuvuuksien avulla.

Markovin satunnaiskenttä on matemaattinen väline, joka tarjoaa keinon mallintaa aineistosta puuttuvaa dataa todennäköisyyslaskennan avulla. Erityisesti joltakin alueelta puuttuvan datan sisältöä voidaan arvioida lähialueiden perusteella. Markovin satunnaiskenttiä käytetään tavallisesti bayesiläisen mallintamisen yhteydessä, erityisesti hierarkkisen mallin osana. Yleensä lähtökohtana on jokin yksinkertainen kuvankäsittelyyn tarkoitettu malli, jota sitten muokataan aineistolle sopivaksi.

Bayesiläisten mallien käsittely on laskennallisesti vaativaa. On käytettävä Markovin ketju Monte Carlo -menetelmää (MCMC), jotta mallien käsittely olisi käytännössä mahdollista. MCMC-menetelmät ovat monipuolisia, mutta samalla laskennallisesti vaativia. Niiden käytössä tulee huomioida useita tekijöitä, jotta tulosten oikeellisuudesta voidaan varmistua. MCMC-menetelmille onkin kehitetty lukuisia erilaisia analyysimenetelmiä ja käytännön toteutuksen ohjeita.

Markovin satunnaiskenttiä ja bayesiläisiä malleja on käytetty myös aiemmin paikkatietoaineistojen puutteiden korjaamiseen. Tässä tutkielmassa niitä käytetään suomen kielen murresana-aineiston ja Suomen lintujen pesimäaineiston yhteydessä. Aineistoille kehitetään yksinkertaiset bayesiläiset todennäköisyysmallit, jotka perustuvat Ising-malliin (eräs Markovin satunnaiskenttä). Molemmat aineistot ovat suuria, joten laskennalliseen tehokkuuteen kiinnitetään erityistä huomiota.

Tutkielman tulokset osoittavat, että suurista paikkatietoaineistoista puuttuvan datan käsittely on aikaa vievää, mutta laskennallisesti mahdollista. Lisäksi aineistoista

puuttuvaa dataa voi mallintaa kohtuullisen hyvin jo yksinkertaisilla Markovin satunnaiskenttiä hyödyntävillä bayesiläisillä malleilla.

Tutkielman luvussa 2 käsitellään sellaisten aineistojen erityispiirteet, joiden havaintoihin voidaan liittää sijaintitietoja. Sen jälkeen luvussa 3 esitellään Markovin satunnaiskenttien teoriaa. Luku 4 sisältää bayesiläisen todennäköisyyslaskennan ja mallintamisen perusteet. Luku 5 käsittää bayesiläisten mallien käsittelyyn tarkoitetun MCMC-menetelmän ja sen analysointitekniikat. Luvussa 6 palautetaan mieleen alan aiempaa tutkimusta ja suoritetaan lopulta testejä oikeille aineistoille. Viimeinen luku 7 tarjoaa yhteenvedon tutkielman sisällöstä.

2 Paikkatietoaineistot

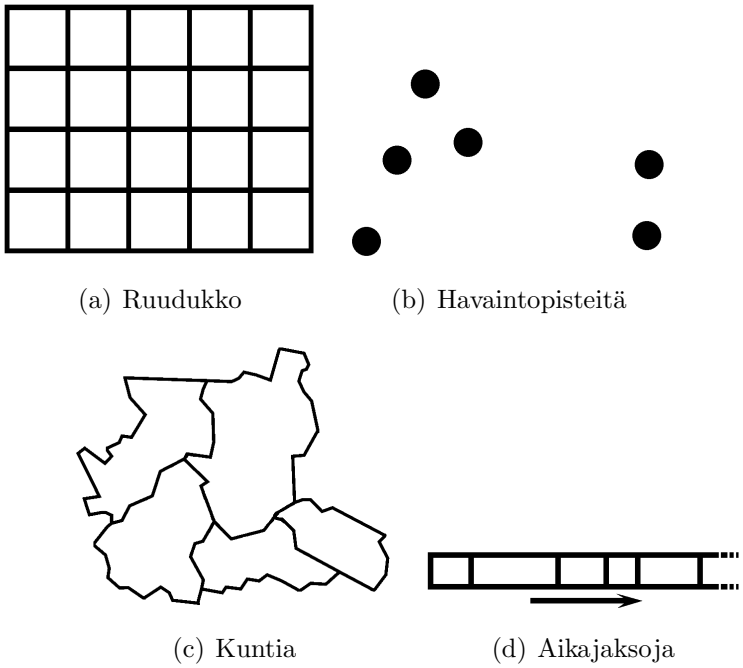
Seuraavaksi käsitellään kerättyjen aineistojen ja erityisesti paikkatietoaineistojen ominaisuuksia. Lisäksi esitellään itsekorrelaation käsite, jonka avulla voidaan mallintaa paikkatietoaineiston ominaisuuksia. Näiden jälkeen pohditaan puutteellisen aineiston käsittelyä.

2.1 Paikkatietoaineistot

Aineistolla tarkoitetaan kokoelmaa dataa, joka on kerätty mistä vain mielenkiinnon kohteena olevasta ilmiöstä. Tarkoitus on hyödyntää kerättyä dataa ja tilastollisia menetelmiä niin, että saadaan tutkittavaa ilmiötä koskevia johtopäätöksiä. Hyvässä aineistossa on niin paljon dataa, että ilmiön tärkeimmät erityispiirteet tulevat esiin, eikä datassa ole juuri (mittaus-)virheitä.

Monien aineistojen sisältämä data voidaan sijoittaa johonkin tiettyyn paikkaan. Tällaisia aineistoja kutsutaan *paikkatietoaineistoiksi* [Goo86], ja ne kuvaavat paikkojen erityispiirteitä. Esimerkkejä paikkatietoaineistojen esitysmuodoista ovat alueruudukko, kartan havaintopisteet ja valtion kunnat. Paikkatietoaineistojen analyysissä tutkitaan usein, millä tavalla maantieteelliset ominaisuudet vaikuttavat tutkittaviin ilmiöihin. Valituilta paikoilta voidaan tehdä havaintoja, jotka kirjataan aineistoon. Esimerkiksi maaston fosforipitoisuuksista on mahdollista muodostaa paikkatietoaineisto.

On myös mahdollista ajatella datan liittyvän johonkin ajanhetkeen aikajanalla, jolloin on kyseessä *aikatietoaineisto*. Niiden analyysissä ollaan yleensä erityisen kiinnostuneita ajan kulumisen aiheuttamista muutoksista tutkittavaan ilmiöön. Esimerkiksi pörssikurssit voivat muodostaa aikatietoaineiston. Tämän tutkielman painotus on paikkatietoaineistoissa, eikä aikatietoaineistoja käsitellä kovin syvällisesti. Aikatietoaineisto voidaan kuitenkin joissakin tapauksissa palauttaa yksiulotteiseksi paikkatietoaineistoksi. Kuvassa 2.1 esitellään erilaisia esitysmuotoja paikkatietoaineiston paikoille ja eräs aikatietoaineiston aikajaksoesitys.



(a) Ruudukko (b) Havaintopisteitä

(c) Kuntia (d) Aikajaksoja

Kuva 2.1: Erilaisia tapoja jaotella havaintohetket ja -paikat.

Datan keräystavat ovat erittäin tilannesidonnaisia. Tärkeintä on varmistaa, että käytetty keräystapa ei vääristä kuvaa tutkittavasta ilmiöstä. Kerääminen on tapahtunut perinteisesti ihmisvoimin. Ihmistyövoiman käyttäminen on kuitenkin taloudellisesti kallista, joten dataa ei ole aina pystytty keräämään kovin paljon.

Erilaisten aineistojen kerääminen on viime vuosikymmeninä helpottunut huomattavasti tietotekniikan käytön lisääntymisen myötä. Joissakin tapauksissa dataa on mahdollista kerätä ohjelmallisesti, jolloin kerääminen on hyvin edullista. Lisäksi nykyinen elektroninen tallennusmuoto on huomattavasti houkuttelevampi kuin aiemmat paperipinot, eikä syyttä: aineistojen käsitteleminen on nyt paljon helpompaa kuin aiemmin.

Lisääntynyt datamäärä on aiheuttanut sen, että kaikkia aineistoja ei ole enää mahdollista käsitellä samoilla tilastollisilla menetelmillä kuin ennen. Varsinkin suurten aineistojen tilastollinen analyysi on käsityönä mahdoton urakka. *Tiedon louhinta* (engl. data mining) on tietojenkäsittelyn ala, jossa pyritään löytämään suurista aineistoista mielenkiintoisia piirteitä automaattisesti. Sen tarkoituksena on automatisoida muun muassa tilastotieteen menetelmiä niin, että suurtenkin aineistojen analyysi olisi tehokasta.

2.2 Itsekorrelaatio

Kun paikkatietoaineistoa tutkitaan tarkasti, siinä voidaan usein havaita säännönmukaisuuksia. Ensivaikutelma on, että toisiaan lähellä tehdyt havainnot eivät olisi toisistaan riippumattomia: monet paikkaan liittyvät havainnot ovat nimittäin varsin samanlaisia kuin lähialueiden havainnot. Kun mitataan esimerkiksi maaston korkeuksia, niin tulokset voidaan osin päätellä lähialueiden korkeuksista. Näennäinen riippuvuus lähialueista onkin yleensä lukuisten vaikuttavien tekijöiden tulos, jotka ovat voimakkaimmillaan lyhyillä etäisyyksillä. Laskennallisista syistä näitä kaikkia todellisia tekijöitä ei voida ottaa huomioon, joten ajatusta lähialueiden vaikutuksesta on yksinkertaistettava. Paikkatietoaineistojen säännönmukaisuudet voidaan monesti selittää sillä, että koko paikka-avaruus vaikuttaa havaintoihin, mutta lähialueet kaikkein voimakkaimmin. Tästä syystä oletetaan, että paikkojen välinen etäisyys riittää kuvaamaan niiden välisen riippuvuuden voimakkuutta.

Edellä kuvattua yksinkertaistusta alueiden vaikutuksesta lähialueisiin kutsutaan tilastotieteessä *itsekorrelaatioksi* (engl. autocorrelation) [Goo86]. Tilastotieteessä on mahdollista laskea paikkatietoaineistolle tunnuslukuja, jotka kuvaavat lähialueiden vaikutuksen voimakkuutta havaintoihin – itsekorrelaation määrää. Kaikille erilaisille laskentatavoille on yhteistä, että aineistossa ei ole itsekorrelaatiota, jos ja vain jos data on riippumaton paikasta. Itsekorrelaation määrän laskeminen tilastotieteen keinoilla vaatii, että aineisto ei saa olla puutteellista – ainakaan tavalla, joka vääristäisi tuloksia. Valitettavasti ihmisten tuottamat aineistot ovat harvoin täydellisiä.

Aikatietoaineistojen kohdalla päädytään samaan: havainnot muistuttavat lähiaikoina tehtyjä havaintoja. Voidaan olettaa, että tällä yksiulotteisella aika-akselilla esiintyy itsekorrelaatiota. Esimerkiksi meriveden lämpötila on tavallisesti pääteltävissä edellisen päivän mittausten perusteella.

Esimerkki 2.2 (Ising-mallin esittely)**.**
Tässä tutkielmassa käydään esimerkkinä läpi yksinkertaistetun Ising-mallin muodostaminen ja soveltaminen todellisiin paikkatietoaineistoihin. Alun perin malli kehitettiin simuloimaan metallin magnetoitumista, mutta sen jälkeen mallille on löytynyt muutakin käyttöä [KS80]. Ising-mallissa tarkasteltavana on kuvan 2.1(a) kaltainen ruudukko, jonka ruudut voivat saada joko arvon 0 *tai* 1*. Ruudukon koko on tässä* 10×10 *ja sen ruutuihin viitataan joukon* $I = \{1, 2, \ldots, 100\}$ *alkioilla. Ideana oli alun perin, että kun metallin lämpötilaa lasketaan, niin yhä suurempi osa metallista magnetoituu samalla tavalla (ruudut muistuttavat viereisiä ruutuja). Lämpötilan laskeminen vastaa ruutujen välisen riippuvuuden kasvamista, mikä johtaa voimakkaaseen itsekorrelaatioon.*

2.3 Aineiston esikäsittely

Monet aineistot ovat jollakin tavoin puutteellisia. Tämä voi johtua siitä, että datan kerääminen on kallista, eikä kaikkea tarpeellista dataa ole saatu kerätyksi. Toisaalta syy voi olla epätarkoissa havainnointivälineissä. Aineiston puutteiden korjaaminen olisi hyödyllistä, sillä jatkoanalyysin tuloksista saadaan tarkempia ja luotettavampia. Uutta dataa ei kuitenkaan aina ole mahdollista kerätä. Yksi keino käsitellä puuttuvaa dataa on yrittää mallintaa sitä jollakin sopivalla mallilla. Parhaassa tapauksessa malli kuvaa ilmiötä niin tarkasti, että sen avulla voidaan luoda hyviä arvioita puuttuvan datan sisällöstä.

Puutteellisen paikkatietoaineiston täydentäminen uskottavalla tavalla on haasteellista. Erityisesti paikkatietoaineistojen yhteydessä on monesti hyödyllistä olettaa itsekorrelaatio: paikkojen välinen etäisyys selittää samankaltaisuuden. Itse asiassa aineiston täydentäminen itsekorrelaatio-oletusta hyödyntävällä mallilla on uskottavampaa kuin pelkkiin paikan ominaisuuksiin perustuva täydentäminen, sillä lähialueidenkin tieto huomioidaan. Itsekorrelaatio-oletusta hyödyntävät mallit suosivat tavallisesti tilanteita, joissa lähekkäisiin paikkoihin liittyvät arvot muodostavat säännönmukaisuuksia.

3 Markovin satunnaiskentät

Jotta paikkatietoaineistoa voitaisiin täydentää ja tarkentaa, tarvitaan menetelmä puuttuvan datan mallintamiseksi itsekorrelaatio-oletuksen avulla. Tätä tarkoitusta varten esitellään Markovin satunnaiskentät. Ensiksi kuvaillaan, miten aineistoa tulee esikäsitellä, jotta Markovin satunnaiskenttiä voidaan käyttää, ja sitten käsitellään aihepiirin teoriaa. Annettuja käsitteitä selvennetään esimerkkien avulla. Ideana on käyttää hyväksi aineiston naapurustorakennetta: Markovin satunnaiskentissä paikkojen ominaisuudet ovat ehdollisesti riippumattomia muiden paikkojen ominaisuuksista, kun naapureiden ominaisuudet tunnetaan.

3.1 Naapuruston määrääminen

Paikkatietoaineistosta puuttuvan datan mallintaminen Markovin satunnaiskentän avulla vaatii, että aineisto esikäsitellään ensin sillä tavalla, että se muodostaa suuntaamattoman verkon. Verkon solmut vastaavat aineiston havaintopaikkoja, jotka on yleensä helppo selvittää aineiston perusteella. Verkon solmuja eli havaintopaikkoja kuvataan suurilla kirjaimilla, sillä solmut rinnastetaan myöhemmin satunnaismuuttujiin.

Määritelmä 3.1 (Naapurusto [Win95, luku 3.1])**.**
Olkoon $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ *suuntaamaton verkko, jossa* $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ *on solmujen joukko ja* $\mathcal{E}$ *solmujen välisten kaarten joukko. Solmujen indeksit ovat joukon* $S = \{1, 2, \ldots, n\}$ *alkioita, jossa* $n = |\mathcal{X}|$ *on solmujen lukumäärä. Kaaret ovat järjestettyjä pareja* (X_i, X_j) *joillakin* $i, j \in S$*. Jos* $X_i, X_j \in \mathcal{X}$ *ja* $(X_i, X_j) \in \mathcal{E}$*, niin solmujen* X_i *ja* X_j *sanotaan olevan* naapureita *(engl. neighbour) keskenään. Oletetaan lisäksi, että mikään solmu ei ole naapuri itsensä kanssa eli* $(X_i, X_i) \notin \mathcal{E}$ *kaikilla* $i \in S$*. Silloin* naapuriverkko $\mathcal{G}$ *määrittelee kaikille solmuille* $X_i \in \mathcal{X}$ naapuruston *(engl. neighbourhood)* $\mathcal{N}(X_i)$*, joka on solmun* X_i *naapurisolmujen joukko. Toisin sanoen,* $\mathcal{N}(X_i) = \{X_j \in \mathcal{X} \mid (X_i, X_j) \in \mathcal{E}\}$*. Suuntaamattomuudesta seuraa, että naapuriominaisuus on symmetrinen:* $X_j \in \mathcal{N}(X_i)$*, jos ja vain jos* $X_i \in \mathcal{N}(X_j)$*.*

Aineiston esikäsittelyvaiheessa toistensa naapureiksi tulisi valita sellaisia solmupareja, joilla voidaan ajatella olevan merkittäviä keskinäisiä riippuvuuksia. Solmun naapurien lukumäärä on tavallisesti melko pieni, yleensä alle kymmenen. Solmun ominaisuudet riippuvat eniten naapureista – riippuvuus muista solmuista välittyy

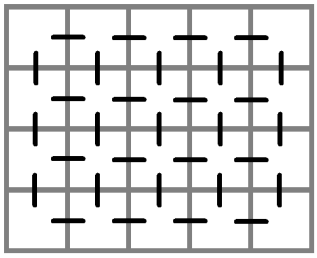
naapurisolmujen kautta. Jos aineistossa oletettu itsekorrelaatio on erityisen monimutkainen, niin naapurien lukumäärän lisääminen voi parantaa tuloksia [TB98].

Aikatietoaineistossa solmujen (aikajaksojen) naapureiksi voidaan asettaa edeltävää ja seuraavaa aikajaksoa kuvaavat solmut. Näin ollen kaikilla solmuilla on kaksi naapuria, paitsi ensimmäisellä ja viimeisellä solmulla (aikajaksolla), joilla on vain yksi naapuri. Solmujen riippuvuuksia ei tulkita tässä kausaalisiksi. Jos paikkatietoaineisto koostuu valtion kunnista, solmun naapureiksi voidaan valita kaikki ne solmuparit (kunnat), joilla on yhteistä rajaa. Tämä soveltuu myös, jos solmut vastaavat ruudukon ruutuja. Kolmiulotteiset ruudukotkaan eivät ole ongelma.

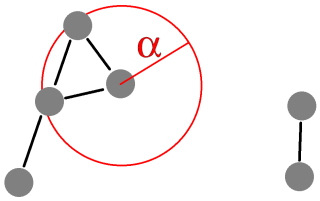
Pisteaineiston naapuruston määrääminen on hankalampaa, mutta siinäkin voidaan asettaa solmut naapureiksi esimerkiksi silloin, kun niiden välinen etäisyys on vähemmän kuin jokin ennalta määrätty kynnysarvo [Li01]. On myös mahdollista käyttää *Voronoi-ositusta* [Aur91], jossa koko pisteavaruus ositetaan solukoksi. Jokaisella solmulla on yksi solu. Pisteavaruuden jokaisen pisteen kohdalla selvitetään, mikä solmuista on sitä lähinnä, ja liitetään piste lähintä solmua vastaavaan soluun. Näin muodostunut solukko osittaa pisteavaruuden mosaiikkimaisiksi palasiksi. Solmut määrätään naapureiksi, jos niitä vastaavilla Voronoi-soluilla on yhteinen raja. Tällaista naapurustorakennetta kutsutaan *Delaunay-naapurustoksi*. Kuvassa 3.3 on esimerkkejä erilaisista naapurustoista.

Esimerkki 3.2 (Ising-mallin naapurusto)**.**
On tavallista asettaa ruudukolle naapurusto, johon kuuluu joko enintään 4 tai enintään 8 naapuria. Tässä siis päätetään, ovatko kulmittain kiinni olevat ruudut toistensa naapureita. Tässä päädytään yksinkertaisempaan ratkaisuun: ruuduilla on 2, 3 tai 4 naapuria (reunaruuduilla on vähemmän naapureita). Ising-mallin naapurusto on identtinen kuvassa 3.3(a) esitetyn kanssa muuten, paitsi että ruudukon koko on 10×10*.*

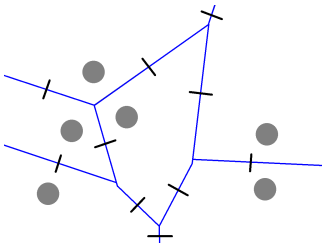
Määritelmän 3.1 mukaiset naapurustot ovat symmetrisiä ja antirefleksiivisiä. Jatkon kannalta on tärkeää tarkistaa, onko näin muodostettu naapuriverkko yhtenäinen. Jos verkon kaksi solmua eivät ole saavutettavissa toisistaan naapurikaaria pitkin, niitä vastaavien muuttujien välillä ei voi riippuvuuksia. Tämä voi olla mallinnuksen kannalta toivottavaa, jos aineisto sisältää toisistaan riippumattomia alueita tai aikajaksoja.



(a) Ruudukon naapurusto

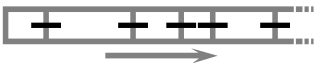


(b) Havaintopisteiden naapurusto kynnysarvolla α



(c) Havaintopisteiden Voronoi-ositus ja Delaunay-naapurusto

(d) Kuntien naapurusto



(e) Aikajaksojen naapurusto

Kuva 3.3: Erilaisia naapurustoja; naapurit on yhdistetty toisiinsa mustalla viivalla.

3.2 Markovin satunnaiskentän määritelmä

Tässä aliluvussa esitellään Markovin satunnaiskenttä, joka voidaan ajatella erityisenä joukkona satunnaismuuttujia. Jatkossa rinnastetaan satunnaismuuttuja ja naapuriverkon solmu toisiinsa. Toisin kuin yksittäisellä satunnaismuuttujalla, satunnaiskentän arvot ovat useiden yksittäisten satunnaismuuttujien arvojen yhdistelmiä, kuten pian nähdään. Näiden satunnaismuuttujien välillä on (syklisiä) riippuvuuksia, jotka saavat aikaan itsekorrelaation esiintymisen.

Määritelmä 3.4 (Satunnaiskenttä [Win95, luku 3.1])**.**
Olkoon $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ jokin verkko ja A_i solmun (satunnaismuuttujan) $X_i \in \mathcal{X}$ mahdollisten arvojen joukko kaikilla $i \in S$. Kutsutaan verkon $\mathcal{G}$ asetelmaksi *vektoria $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$, jossa $a_i \in A_i$ kaikilla $i \in S$. Asetelma sisältää arvot jokaiselle verkon $\mathcal{G}$ solmulle. Kokoelma $\mathcal{A} = \{\boldsymbol{a} \mid a_i \in A_i\}$ sisältää verkon $\mathcal{G}$ kaikki mahdolliset asetelmat. Kokoelman $\mathcal{A}$ asetelmien ja niiden todennäköisyyskuvauksen* PR *yhdistelmää kutsutaan* satunnaiskentäksi *(engl. random field), jos* PR *on aidosti positiivinen ja integroituu ykköseksi* [1]*. Toisin sanoen, $\text{PR}(\boldsymbol{a}) > 0$ kaikilla asetelmilla $\boldsymbol{a} \in \mathcal{A}$ ja lisäksi $\int_{\boldsymbol{a} \in \mathcal{A}} \text{PR}(\boldsymbol{a})\, d\boldsymbol{a} = 1$. Jos satunnaismuuttujien mahdollisten arvojen joukko on diskreetti, niin todennäköisyyksien tulee summautua ykköseksi.*

Oletetaan että satunnaiskentän solmuja vastaavat satunnaismuuttujat antavat kukin arvolleen aidosti positiivisen todennäköisyystiheyden. Silloin määritelmän 3.4 mukaan myös todennäköisyyskuvaus PR antaa aidosti positiivisen todennäköisyystiheyden asetelmalle, jossa solmuilla on nämä arvot. Kuvaus PR voikin antaa todennäköisyystiheydeksi nollan, jos ja vain jos satunnaiskentän jonkin solmun (satunnaismuuttujan) arvo on mahdoton.

Määritelmä 3.5 (Markovin satunnaiskenttä [Win95, Bes74, KS80])**.**
Olkoon $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ naapuriverkko. Olkoon lisäksi mahdollisten asetelmien kokoelma $\mathcal{A}$ ja niiden todennäköisyyskuvaus PR *yhdessä verkkoon liittyvä satunnaiskenttä. Tätä satunnaiskenttää kutsutaan* Markovin satunnaiskentäksi *(engl. Markov random field, Markov network) naapuriverkolle $\mathcal{G}$, jos kaikilla $i \in S$ pätee ns. Markovin ehto*

$$\text{PR}(X_i \mid \mathcal{X} \setminus \{X_i\}) = \text{PR}(X_i \mid \mathcal{N}(X_i)).$$

[1] Diskreettien todennäköisyyskuvausten arvot ovat aina enintään yksi, mutta jatkuvilla kuvauksilla todennäköisyystiheydet voivat olla myös ykköstä suurempia. Esimerkiksi jatkuvan todennäköisyyskuvauksen f todennäköisyystiheydet ovat sen kertymäfunktion $F(x) = \text{PR}(X \leqslant x)$ derivaattoja $f(x) = \frac{\partial\, F(x)}{\partial\, x}$. Jos todennäköisyyskuvaus on diskreetti, niin tässä tutkielmassa käytetään kuvauksesta merkintää Pr; jatkuvissa tapauksissa (ja oletuksena) käytetään merkintää PR.

Määritelmässä 3.5 esiintynyt Markovin ehto tarkoittaa, että minkä tahansa solmun (satunnaismuuttujan) arvon todennäköisyystiheys ehdollistettuna kaikkien muiden solmujen arvoilla on sama kuin todennäköisyystiheys samalle arvolle ehdollistettuna vain naapurisolmujen arvoilla. Edellisen voi tulkita niin, että kaikkien muiden satunnaismuuttujien arvojen vaikutus kätkeytyy naapureiden arvoihin. Tämä on Markovin satunnaiskenttien merkittävin etu laskennallisessa mielessä: simulointiohjelma tutkii vain naapureiden arvoja ja jättää muiden solmujen arvot huomiotta.

Määritelmän 3.5 mukaisissa todennäköisyyskuvauksissa solmujen arvojen ehdolliset todennäköisyystiheydet ovat syklisiä, eivätkä kaikki kuvaukset tästä syystä välttämättä toteuta todennäköisyyskuvauksen määritelmää. Onneksi paljastuu, että Markovin satunnaiskentät voidaan ilmaista myös seuraavalla, usein helpommalla tavalla.

Määritelmä 3.6 (Gibbsin satunnaiskenttä [Win95, Bes74, KS80])**.**
Olkoon $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ naapuriverkko. Epätyhjä solmujoukko $C \subseteq \mathcal{X}$ on klikki *(engl. clique, simplex) tai* täydellinen osaverkko*, jos sen kaikki solmuparit ovat keskenään naapureita eli kaikilla solmupareilla $X_i, X_j \in C$ pätee $(X_i, X_j) \in \mathcal{E}$, kun $i \neq j$. Myös yksittäiset solmut ovat klikkejä. Olkoon $\mathcal{C}$ naapuriverkon $\mathcal{G}$ kaikkien klikkien kokoelma, $\mathcal{C} = \{\emptyset \subset C \subseteq \mathcal{X} \mid \forall X_i, X_j \in C : i \neq j \Rightarrow (X_i, X_j) \in \mathcal{E}\}$ (katso kuva 3.7). Naapuriverkon $\mathcal{G}$ asetelmien kokoelma $\mathcal{A}$ ja niiden todennäköisyyskuvaus*

$$\textsc{pr}(\boldsymbol{a}) = \frac{1}{Z} \cdot \exp\left(\sum_{C \in \mathcal{C}} \Phi_C(\boldsymbol{a})\right)$$

yhdessä määrittävät Gibbsin satunnaiskentän *(engl. Gibbs random field) naapuriverkolle $\mathcal{G}$. Todennäköisyyskuvauksen* PR *termi*

$$Z = \int_{\boldsymbol{b} \in \mathcal{A}} \exp\left(\sum_{C \in \mathcal{C}} \Phi_C(\boldsymbol{b})\right) \, d\boldsymbol{b}$$

on normalisointivakio*, joka ei riipu asetelmasta $\boldsymbol{a}$. Jos arvojoukko on diskreetti, niin integraali korvataan summalla. Funktiota Φ_C kutsutaan klikin C* naapuripotentiaalifunktioksi *naapuriverkossa $\mathcal{G}$. Sen arvo saa riippua erityisesti klikkiin C kuuluvien solmujen ominaisuuksista ja niiden arvoista, mutta ei mistään muista solmuista.*

Gibbsin satunnaiskenttiä on huomattavasti helpompi muodostaa kuin Markovin satunnaiskenttiä. Malleihin sopivien potentiaalifunktioiden valinta ei valitettavasti ole aivan suoraviivaista, mutta funktion suuret arvot kertovat, että klikin solmuilla on mallintamisen kannalta uskottavat arvot, esimerkiksi kaikki arvot ovat samanlaisia.

Kuva 3.7: Kuntien naapuriverkon klikit: pisteet, viivat ja kolmiot vastaavat yhden, kahden ja kolmen kokoisia klikkejä.

Yksi potentiaalifunktioiden valintatapa voisi olla seuraava. Määrätään aluksi solmujen arvoille haluttuja todennäköisyystiheyksiä ehdollistettuna valituilla naapuriarvoilla. Tämän jälkeen kehitetään sellaiset potentiaalifunktiot, jotka tuottavat halutut todennäköisyystiheydet. Potentiaalifunktiot määräävät satunnaiskentän käyttäytymisen klikkitasolla, joten koko satunnaiskentän alueelle ulottuvaa vaikutusta on joskus hankala ennustaa pelkkien potentiaalifunktioiden avulla. Vaikka täsmällinen tulkinta puuttuukin, potentiaalifunktioiden karkea vaikutus on silti useimmiten pääteltävissä. Ei voida kuitenkaan sanoa, että tietyt potentiaalifunktiot aiheuttaisivat juuri tietyn suuruista itsekorrelaatiota koko satunnaiskentän alueella.

Määritelmässä 3.6 esiintynyttä todennäköisyyskuvausta on mahdollista yksinkertaistaa. Jos naapuriverkko on *tasoittuva* (engl. planar) [Bol98] eli se voidaan piirtää ilman risteäviä kaaria, niin siinä voi olla korkeintaan neljän solmun kokoisia klikkejä. Potentiaalifunktioiden määrää voidaan rajoittaa myös asettamalla sama potentiaalifunktio kaikille samankokoisille klikeille. On myös tavallista määrätä niin, että suurempien kuin kahden kokoisten klikkien potentiaalifunktiot ovat identtisesti nollia ja jäävät huomiotta [Bes74]. Tällainen suurien klikkien sivuuttaminen soveltuu silloin, kun solmujen väliset riippuvuudet halutaan säilyttää yksinkertaisena. Monimutkaisempien vuorovaikutusten tapauksessa voidaan vastaavasti jättää huomiotta pienimmät klikit ja keskittyä vain maksimaalisiin klikkeihin [TB98]. Erilaisten potentiaalifunktioiden määrää voidaan siis rajoittaa, jotta laskentatyö keventyisi.

Gibbsin satunnaiskentän määritelmän 3.6 todennäköisyyskuvauksessa esiintyvä normalisointivakio Z on laskennallisesti hyvin ongelmallinen. Tarkastellaan varsin yksinkertaista satunnaiskenttää, jossa on 100 solmua ja kaikilla solmuilla on kaksi mah-

dollista arvoa (esimerkiksi Ising-malli). Erilaisia asetelmia on $|\mathcal{A}| = 2^{100}$ kappaletta, joten normalisointivakion tarkan arvon laskeminen yksinkertaisilla algoritmeilla on käytännössä mahdotonta. Ising-mallin joidenkin erityistapauksien normalisointivakio on silti onnistuttu selvittämään polynomisessa ajassa toimivalla, joskin hankalalla, algoritmilla [GLV00]. Tavallisesti kuvauksen PR tarkat todennäköisyystiheydet eivät ole käytettävissä. Myöhemmin näytetään, kuinka tämä ongelma voidaan kiertää laskemalla kuvauksen PR ehdollisia todennäköisyystiheyksiä.

Lause 3.8 (Hammersley–Clifford [Win95, luku 3.3])**.**
Olkoon $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ naapuriverkko. Satunnaiskenttä on Markovin satunnaiskenttä naapuriverkolle $\mathcal{G}$, jos ja vain jos se on Gibbsin satunnaiskenttä samalle naapuriverkolle.

Lause 3.8 on todistettu myös useissa muissa yhteyksissä [Bes74, Pre74, KS80]. Sen seurauksena jokainen Gibbsin satunnaiskenttä toteuttaa Markovin ehdon. Haluttu Markovin satunnaiskenttä on siis mahdollista muotoilla helpommalla tavalla määritelmän 3.6 mukaisesti. Jatkossa muodostetaan satunnaiskenttiä Gibbsin määritelmän mukaisesti ja niistä käytetään nimeä Markovin satunnaiskenttä.

Esimerkki 3.9 (Markovin satunnaiskenttä Ising-mallille)**.**
Ising-mallin jokaisen ruudun mahdollisten arvojen joukko on $A_i = \{0, 1\}$ kaikilla $i \in I$ ja asetelmien kokoelma on $\mathcal{A} = \{0, 1\}^{100}$. Asetetaan Ising-mallille seuraavaksi määritelmän 3.6 mukaiset potentiaalifunktiot. Alkuperäisen mallin potentiaalifunktiosta on tässä jätetty pois joitakin fysiikkaan liittyviä muuttujia, jotta funktiosta tulisi yksinkertaisempi. Ruudukossa, jossa ruudulla on enintään 4 naapuria, on vain yhden ja kahden kokoisia klikkejä. Potentiaalifunktioita Φ_i käytetään yhden ruudun kokoisiin klikkeihin ja potentiaalifunktioita $\Phi_{i,j}$ kahden ruudun kokoisiin. Määritellään funktiot

$$\Phi_i(\boldsymbol{a}) = \alpha_i \cdot a_i,$$

$$\Phi_{i,j}(\boldsymbol{a}) = \beta_{i,j} \cdot (a_i \cdot a_j + (1 - a_i) \cdot (1 - a_j)) = \beta_{i,j} \cdot 1_{a_i = a_j},$$

jossa $\alpha_i, \beta_{i,j} \in \mathbb{R}$. Intuitiivisesti tulkittuna: Jos ruutukohtainen parametri α_i on positiivinen, niin funktio Φ_i suosii arvoa 1 arvon 0 kustannuksella ruudussa $i \in I$. Negatiivisella arvolla suositaankin arvoa 0. Jos $\beta_{i,j}$ on positiivinen, niin funktio $\Phi_{i,j}$ suosii tilanteita, joissa klikin molemmilla ruuduilla i ja j on sama arvo. Jos taas $\beta_{i,j}$ on negatiivinen, niin ruuduissa i ja j suositaan toisistaan poikkeavia arvoja. Tapauksessa $\beta_{i,j} = 0$ potentiaalifunktioilla $\Phi_{i,j}$ ei ole vaikutusta eli ruutujen i ja j välillä ei ole suoraa riippuvuutta.

Kun $\mathcal{C}_1$ ja $\mathcal{C}_2$ sisältävät Ising-ruudukon kaikki yhden ja kahden kokoiset klikit, niin Markovin satunnaiskentälle saadaan todennäköisyyskuvaus

$$\Pr(\boldsymbol{a}) = \frac{1}{Z} \cdot \exp\left(\sum_{\{X_i\} \in \mathcal{C}_1} \alpha_i \cdot a_i + \sum_{\{X_i, X_j\} \in \mathcal{C}_2} \beta_{i,j} \cdot 1_{a_i = a_j} \right).$$

Normalisointivakion Z laskeminen todettiin jo aiemmin käytännössä hankalaksi.

Joskus Markovin satunnaiskentän satunnaismuuttujille halutaan asettaa jatkuva arvojoukko. Silloin todennäköisyyskuvauksen PR on oltava integroituva. Lisäksi määritelmän 3.6 potentiaalifunktiot täytyy valita niin, että normalisointivakio Z pysyy äärellisenä. Kun nämä ehdot on täytetty, niin jatkuvien arvojoukkojen käyttäminen on sallittua [Bes74].

Jos satunnaismuuttujilla on jatkuva arvojoukko, niin voidaan käyttää *normaalijakautuneita Markovin satunnaiskenttiä* (engl. Gaussian Markov random field) [Bes75, RH05]. Silloin Markovin satunnaiskenttä muodostaa moniulotteisen normaalijakauman ja ehdolliset jakaumat ovat nekin normaalisia. Tämä sopii erityisen hyvin solmujen arvoissa esiintyvien mittausvirheiden mallintamiseen. Tämän tutkielman pääpaino on kuitenkin diskreetissä Ising-mallissa.

Markovin satunnaiskenttien merkittävin ominaisuus on, että niiden avulla on mahdollista muodostaa eheä todennäköisyyskuvaus (syklisten) satunnaismuuttujien arvoyhdistelmille. Siksi niitä voidaan käyttää luontevasti mukana todennäköisyyksiin perustuvassa mallintamisessa. Suurin yksittäinen heikkous on, että sopivien potentiaalifunktioiden keksiminen voi olla hankalaa – ainakin silloin, kun potentiaalifunktio kuvaa monimutkaista ilmiötä.

3.3 Sovelluksia

Markovin satunnaiskenttiä voidaan käyttää lukuisiin erilaisiin tarkoituksiin, joissa halutaan hyödyntää lähialueiden välisiä riippuvuuksia. Naapurisolmujen välinen riippuvuus ilmaistaan sopivilla potentiaalifunktioilla. Yksinkertaisetkin mallit ovat monesti soveliaita monien aineistojen käsittelyyn.

Valtaosa Markovin satunnaiskenttien sovelluksista on alun perin kehitetty kuvankäsittelyyn, mutta samat menetelmät yleistyvät usein myös muille aloille [Li01]. Tutkielmassa tullaan luvussa 6 näyttämään, miten kuvankäsittelyyn suunnattuja

Markovin satunnaiskenttien sovelluksia voidaan hyödyntää myös todellisten paikkatietoaineistojen puutteiden mallintamisessa. Seuraavaksi esitellään joitakin yksinkertaisia Markovin satunnaiskenttien kuvankäsittelysovelluksia.

Kuvat koostuvat pikseleistä, jotka ovat järjestäytyneet ruudukoksi. Tämä tarjoaa valmiin naapurustorakenteen Markovin satunnaiskentille. Kuvat voidaan jakaa karkeasti kahteen luokkaan niiden värityksen perusteella. Jos kuvassa on vain muutama eri väri (tai sävyero), kuvan väritystä kutsutaan *diskreetiksi*. Toinen vaihtoehto on, että kuvan värimaailma on *jatkuva*: kuvan väriarvot ilmaistaan reaalilukuarvona.

Kuvien *korjaamiseksi* [Li01, Win95] kutsutaan sellaista kuvan muokkaamista, jossa vahingoittunutta kuvaa muokataan niin, että se muistuttaisi paremmin alkuperäistä kuvaa. Kuvaan on esimerkiksi voitu lisätä satunnaista kohinaa, jolloin joidenkin pikseleiden väriarvo muuttuu. Markovin satunnaiskentät soveltuvat sekä diskreetin että jatkuvan värimaailman kuvien korjaamiseen, jos kohinaa ei ole kovin paljon. Jatkuvan värimaailman kuvia on myös mahdollista *silottaa* (engl. smoothing). Silottamisessa kuvien voimakkaita värimuutoksia pyritään pehmentämään lisäämällä muutosalueille värien sekoituksia.

Diskreetit kuvat ovat yksinkertaisia, ja siksi niille on kehitetty paljon erilaisia sovelluksia. Eräs vanhimmista on *rajojen tunnistaminen* [GG84, Win95], jossa kuvasta etsitään kapeita nauhamaisia alueita, joiden eri puolien värit poikkeavat toisistaan voimakkaasti. Menetelmää voi käyttää esimerkiksi tunnistamaan kuvista esineiden ääriviivoja. Kuvien *lohkomisessa* (engl. segmentation) [Li01] taas pyritään päinvastaiseen: tarkoitus on erottaa kuvasta väreiltään yhtenäisiä lohkoja. Näin löydetyt lohkot voidaan usein tulkita esineiksi tai erillisiksi alueiksi.

4 Bayesiläinen mallintaminen

Seuraavaksi esitellään lyhyesti bayesiläisen todennäköisyyslaskennan perusteet. Niiden jälkeen tutustutaan bayesiläiseen mallintamiseen ja suosittuun mallintamistekniikkaan: hierarkkisiin malleihin. Lopuksi esitellään erilaisia tapoja hyödyntää Markovin satunnaiskenttiä hierarkkisissa todennäköisyysmalleissa.

4.1 Bayes-päättelyn perusteet

Todennäköisyyksien tulkinnassa on kaksi erilaista koulukuntaa: frekventistinen ja bayesiläinen [Nea93, s. 4][GCSR04, luku 1.5]. Eri koulukuntien puolustajilla on vankkoja käsityksiä omien kantojensa paremmuudesta. Tässä tutkielmassa esitetään lyhyesti molemmat kannat, ja jatkossa käytetään bayesiläistä tulkintaa.

Frekventistisessä (perinteisessä) todennäköisyystulkinnassa todennäköisyysarvot mittaavat tapahtumien esiintymistiheyksiä, kun koetilannetta toistetaan lukuisia kertoja. Nopan silmäluvulle kaksi voidaan asettaa todennäköisyys $\frac{1}{6}$, sillä kakkosten määrä on pitkässä toistosarjassa kuudesosa heittojen lukumäärästä. Tulkinta ei sovellu tilanteisiin, joissa tilannetta ei voida ajatella toistettavan uudelleen. Esimerkiksi jokin historiallinen mennyt tapahtuma joko tapahtui tai sitten ei: jo tapahtuneeseen ei voi enää liittyä mitään satunnaisuutta, koska arvo on jo kiinteä. Frekventistisen todennäköisyyden voikin liittää vain satunnaisiin ilmiöihin, joiden lopputulos ei ole vielä kiinnitetty.

Bayesiläisessä tulkinnassa todennäköisyys tulkitaan vapaammin subjektiivisen uskomuksen mitaksi. Tärkein ero frekventistiseen näkemykseen on, että tarkkailtavien tapahtumien ei vaadita olevan aidosti satunnaisia. Oletetaan että tutkittava arvo on tuntematon subjektiiviselle tarkkailijalle. Silloin tarkkailijan epävarmuus todellisesta arvosta voidaan ilmaista bayesiläisittäin todennäköisyysjakauman avulla. Frekventistien mielestä kiinteiden arvojen kohtelu (uskomuksen) satunnaismuuttujina on perustelematonta ja vastoin intuitiota. Frekventistisen toistokokeen sijaan korostetaankin subjektiivista todennäköisyyden tulkintaa. Bayesiläinen tulkinta edellä mainitusta nopanheitosta on, että subjektiivinen tarkkailija uskoo yhden nopanheiton silmäluvun olevan kaksi varmuudella $\frac{1}{6}$. Vain bayesiläinen voi sanoa historiallisen menneen tapahtuman todennäköisyyden olevan vaikkapa 0,87, jolloin hän uskoo enemmän tapahtumiseen kuin ei-tapahtumiseen. Bayesiläinen todennäköisyystulkinta on siis laajempi, mutta vaatii bayesiläisen ajattelutavan hyväksymistä.

Tarkastellaan todennäköisyysmallia $\mathcal{M}$, ja sen sisältämien satunnaismuuttujien joukkoja Θ ja $\mathcal{D}$. Jatkossa rinnastetaan mallin *muuttujat* ja satunnaismuuttujat. Bayesiläisessä data-analyysissä mallin $\mathcal{M}$ kaikille muuttujille määritellään *yhteisjakauma* (engl. joint probability distribution) [GCSR04, luku 1.3]

$$\text{PR}(\boldsymbol{\theta}, \mathcal{D}). \tag{4.1}$$

Yhteisjakauma määrää todennäköisyystiheydet muuttujien Θ ja $\mathcal{D}$ arvojen kaikille

yhdistelmille. Muuttujien $\mathcal{D}$ arvot annetaan yleensä datana. Kun yhteisjakauma ehdollistetaan tällä datalla, niin saadaan tärkeä *Bayesin kaava*:

$$\text{PR}(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{\text{PR}(\boldsymbol{\theta}, \mathcal{D})}{\text{PR}(\mathcal{D})} = \frac{\text{PR}(\boldsymbol{\theta}) \cdot \text{PR}(\mathcal{D} \mid \boldsymbol{\theta})}{\text{PR}(\mathcal{D})} = \frac{\text{PR}(\boldsymbol{\theta}) \cdot \text{PR}(\mathcal{D} \mid \boldsymbol{\theta})}{\int_{\boldsymbol{\theta'}} \text{PR}(\boldsymbol{\theta'}) \cdot \text{PR}(\mathcal{D} \mid \boldsymbol{\theta'}) \, d\boldsymbol{\theta'}}. \quad (4.2)$$

Kaavassa (4.2) esiintyvää termiä $\text{PR}(\boldsymbol{\theta})$ kutsutaan muuttujien arvovektorin $\boldsymbol{\theta}$ *prioritodennäköisyystiheydeksi*, termiä $\text{PR}(\mathcal{D} \mid \boldsymbol{\theta})$ datan $\mathcal{D}$ *uskottavuudeksi* (engl. likelihood) muuttujien arvovektorilla $\boldsymbol{\theta}$ ja termiä $\text{PR}(\boldsymbol{\theta} \mid \mathcal{D})$ muuttujien arvovektorin $\boldsymbol{\theta}$ *posterioritodennäköisyystiheydeksi* datalla $\mathcal{D}$. Lisäksi termiä $\text{PR}(\mathcal{D})$ kutsutaan joskus *datan* $\mathcal{D}$ *reunatodennäköisyystiheydeksi* (engl. marginal probability).

Ajatuksena on, että vektorin $\boldsymbol{\theta}$ muuttujista on annettu jotakin etukäteistietoa priorijakauman muodossa. Uskottavuus kertoo, kuinka todennäköistä on, että juuri muuttujien arvovektorin $\boldsymbol{\theta}$ sisältämät arvot olisivat tuottaneet muuttujille $\mathcal{D}$ datasta saadut arvot. Muuttujien arvovektorin $\boldsymbol{\theta}$ posterioritodennäköisyystiheydet perustuvat sekä etukäteistietoon että annettuun dataan. Posteriorijakauma kertoo, kuinka uskomuksemme muuttuu nähtyämme datan.[2]

Jos muuttujat voivat saada vain diskreettejä arvoja, niin kaavan (4.2) jakajan integraali yksinkertaistuu summaksi. Joka tapauksessa jakaja on riippumaton parametrina annetuista arvoista $\boldsymbol{\theta}$. Usein käytetäänkin yhtäsuuruuden asemesta *verrannollisuutta*

$$\text{PR}(\boldsymbol{\theta} \mid \mathcal{D}) \propto \text{PR}(\boldsymbol{\theta}) \cdot \text{PR}(\mathcal{D} \mid \boldsymbol{\theta}).$$

Verrannollisuutta käyttämällä saadaan arvoja, jotka poikkeavat todellisista posterioritodennäköisyystiheyksistä vakiokertoimella – tarkemmin sanottuna kaavan (4.2) jakajan käänteisluvun verran.

Bayesiläinen data-analyysi pohjautuu posterioritodennäköisyystiheyksien selvittämiseen ja posteriorijakauman ominaisuuksien tutkimiseen. Posteriorijakauman analyyttinen johtaminen on usein ylivoimaista johtuen kaavan (4.2) jakajassa olevasta integraalista, mutta posteriorijakaumaa on silti mahdollista approksimoida MCMC-menetelmien avulla (esitellään luvussa 5.4.2). Menetelmä perustuu siihen, että todennäköisyystiheyksien suhteet ovat selvitettävissä verrannollisuuden avulla – ilman hankalan jakajavakion laskemista.

[2] Apriorinen tieto ja aposteriorinen tieto ovat alun perin filosofisen tietoteorian käsitteitä. Immanuel Kant esitti, että on olemassa tietoa a priori, joka on havainnoista riippumatonta. Samaten on olemassa tietoa a posteriori, joka saadaan havaintojen avulla. Bayesiläisen todennäköisyyslaskennan avainkäsitteet perustuvat näihin termeihin.

On tavallista, että kaikki muuttujat sisältävän posteriorijakauman sijasta ollaan kiinnostuneita vain muutaman muuttujan jakaumasta. Silloin halutaan tarkastella posteriorijakauman *reunajakaumia* (engl. marginal distribution) [GCSR04, luku 3.1], jotka saadaan integroimalla posteriorijakaumasta loput (ei-kiinnostavat) muuttujat pois. Jaetaan arvovektori $\boldsymbol{\theta}$ kahteen osaan, jossa $\boldsymbol{\theta}_+$ on kiinnostavien ja $\boldsymbol{\theta}_-$ on ei-kiinnostavien muuttujien arvovektori. Silloin kiinnostavien muuttujien arvovektorille $\boldsymbol{\theta}_+$ saadaan reunatodennäköisyystiheys

$$\text{PR}(\boldsymbol{\theta}_+ \mid \mathcal{D}) = \int_{\boldsymbol{\theta}_-} \text{PR}(\boldsymbol{\theta} \mid \mathcal{D})\ d\boldsymbol{\theta}_- = \int_{\boldsymbol{\theta}_-} \text{PR}(\boldsymbol{\theta}_- \mid \mathcal{D}) \cdot \text{PR}(\boldsymbol{\theta}_+ \mid \boldsymbol{\theta}_-, \mathcal{D})\ d\boldsymbol{\theta}_-.$$

Kuten posteriorijakauman selvittäminen, myös reunajakaumien selvittäminen on laskennallisesti vaativaa, sillä se vaatii yleensä moniulotteista integrointia.

Bayesiläisellä todennäköisyyslaskennalla on vahva matemaattinen pohja, mutta ongelmiakin on. Bayesiläistä todennäköisyyslaskentaa ei nimittäin voi soveltaa ilman priorijakaumia. Tämä on toisaalta vahvuus ja toisaalta heikkous. Jos muuttujien arvoista on jotakin etukäteistietoa, niin priori sallii keinon vaikuttaa lopputulokseen eli posteriorijakaumaan. Vaikka muuttujien arvoista ei ole erityistä etukäteistietoa, Bayesin kaava vaatii silti priorijakauman asettamista. Valitettavasti on epäselvää, millainen priorijakauma kuvaisi parhaiten täydellistä tietämättömyyttä muuttujan mahdollisista arvoista. Joka tapauksessa, kaiken kattavaa *ei-informatiivista* priorijakaumaa ei ole olemassa: ensivaikutelman vastaisesti myös tasainen jakauma ottaa kantaa muuttujan arvoihin. Priorilla on siis subjektiivinen vaikutus posteriorijakaumaan, kuten bayesiläisellä tulkinnalla todennäköisyyksiin.

4.2 Bayesiläinen hierarkkinen mallintaminen

Mallintamisessa on ajatuksena kuvata tarkkailtavaa ilmiötä mahdollisimman tarkasti jonkin mallin avulla. Parhaassa tapauksessa malli sisältää kaikki tärkeimmät ilmiöön vaikuttavat tekijät. Hyvän mallin avulla on mahdollista selvittää erilaisten tekijöiden vaikutusten voimakkuutta tarkkailtavaan ilmiöön ja ennustaa tulevia tapahtumia.

On harvinaista, että tosielämän ilmiölle löytyy jokin täydellisen hyvä malli. Esimerkiksi monet fysiikan kaavat eli luonnonlakien mallit olettavat jonkin ideaalisen ominaisuuden, esimerkiksi tyhjiön tai aineen tasakoosteisuuden. Onkin sanottu, että

kaikki mallit ovat virheellisiä, mutta jotkut hyödyllisiä. Joka tapauksessa mallin valinta vaatii asiantuntijatietoa mallinnettavasta ilmiöstä. Tavallisinta on, että asiantuntijat tarkentavat mallia vähitellen, kunnes sen ennustustarkkuus ei enää parane. Malleja (esimerkiksi tärkeitä Bayes-verkkoja) on myös mahdollista oppia suoraan datasta koneoppimisen keinoin [Hec97, MSTU02].

Toisaalta mallista halutaan mahdollisimman tarkka ja toisaalta sen parametrien arvoja on voitava arvioida laskennallisesti. Monimutkaiset mallit voivat olla laskennallisesti kestämättömiä. Esimerkiksi sään ennustaminen tarkasti viikolla eteenpäin vaatii niin monimutkaisen mallin, etteivät tulokset voi millään valmistua ajoissa (jos koskaan). Sään ennustamisessa joudutaankin tyytymään yksinkertaisempiin malleihin, joiden antamat ennusteet ovat epävarmempia. Jos alan asiantuntijat voivat hyväksyä yksinkertaisen mallin vaatimat oletukset ja mallin ennusteet vastaavat havaintoja, niin mallia voidaan pitää uskottavana käsityksenä mallinnettavasta ilmiöstä.

Bayesiläistä mallintamista voidaan soveltaa lukuisissa erilaisissa tilanteissa [GCSR04]. Mallit kannattaa muodostaa niin, että niiden käsittely on laskennallisesti tehokasta. Kun mallin muuttujilla on vain vähän riippuvuuksia keskenään, niin mallin yhteisjakauma on rakenteeltaan yksinkertainen ja laskennallisesti helppo. Muuttujien väliset riippuvuudet muodostavat silloin riippuvuushierarkian, jossa ylimmän hierarkiatason muuttujat vaikuttavat alempien hierarkiatasojen muuttujiin, jotka vuorostaan vaikuttavat yhä alempien hierarkiatasojen muuttujiin. Kun muuttujaan θ suoraan vaikuttavien muuttujien arvot on annettu, niin muuttuja θ on *ehdollisesti riippumaton* kaikista muista muuttujista. Tällaisen riippuvuushierarkian sisältävä malli on usein laskennallisesti tehokas ja sille löytyy siisti graafinen esitys.

Bayesiläiset mallit muodostetaan usein *hierarkkisen mallintamisen* [GCSR04, luku 5] avulla. Olkoon $\mathcal{V}$ hierarkkisen mallin $\mathcal{M}$ kaikkia muuttujia vastaavien *solmujen* joukko. Hierarkkisessa mallissa solmut asetetaan eri hierarkiatasoille: hierarkiatason solmut vaikuttavat alemman hierarkiatason solmuihin. Hierarkkinen malli voidaan kuvata solmujen $\mathcal{V}$ ja niiden välisten vaikutuskaarten muodostamana suunnattuna ja syklittömänä verkkona, jonka korkeimmalla hierarkiatasolla sijaitsevat ylimmän tason muuttujat, *hyperparametrit*, ja pohjalla on ainakin osa sellaisista muuttujista, joiden arvot annetaan datana. Kaaret ovat suunnattuja ja kuvaavat solmujen vaikutusta alemman tason solmuihin. Solmu on toisen solmun *vanhempi*, jos edellisestä on suunnattu kaari jälkimmäiseen. Solmun *lapsi* määritellään toisin päin.

Hierarkkiset mallit sisältävät aina ehdollisen riippumattomuuden: solmun arvon todennäköisyystiheys ehdollistettuna vain sen vanhempien arvoilla on sama kuin sen todennäköisyystiheys ehdollistettuna hierarkiassa kaikkien ylempänä olevien solmujen arvoilla. Solmun vanhemmat määrittävät sen arvoille prioritodennäköisyystiheydet ja lapset uskottavuudet. Vain solmun vanhemmilla ja lapsilla on vaikutusta sen arvoihin – muut solmut vaikuttavat siihen vanhempien ja lasten välityksellä.

Hierarkkisten mallien solmujen väliset riippuvuudet (kaaret) ilmaistaan niin, että solmun vanhempien arvoilla on suora vaikutus sen arvojen todennäköisyysjakaumaan. Solmua vastaavalla muuttujalla on priorijakauma, jonka parametreja vanhemmat ovat. Jokaiselle solmulle on asetettava priorijakauma. Jos solmulla on vanhempia, niin priorijakauman parametrit määräytyvät vanhempien arvojen mukaan; muuten priorijakauman parametrit ovat vakioita. Erikoistapauksena on malliin kuuluva vakioarvo, joka ei ole satunnaismuuttuja. Tällaiselle solmulle voidaan asettaa sellainen priorijakauma, jonka arvoalue koostuu yhdestä vakioarvosta, ja tällä arvolla on todennäköisyys yksi.

Olkoon $\mathcal{V}$ jälleen mallin $\mathcal{M}$ kaikkia muuttujia vastaavien solmujen joukko. Mallin yhteisjakauma on tulo mallin solmujen $\mathcal{V}$ arvojen todennäköisyystiheyksistä ehdollistettuna (mahdollisten) vanhempien arvoilla. Vanhemmattomat solmut tuottavat todennäköisyystiheyden omasta kiinteästä priorijakaumastaan – vakion tapauksessa ykkösen. Hierarkkisen mallin $\mathcal{M}$ yhteisjakauma on kokonaisuudessaan

$$\text{PR}(\mathcal{V}) = \prod_{V \in \mathcal{V}} \text{PR}(V \mid \texttt{vanhemmat}(V)). \tag{4.3}$$

Joillakin priorijakaumilla on sellainen ominaisuus, että sopivalla uskottavuustodennäköisyystiheydellä kerrottuna ne kuuluvat edelleen samaan jakaumaperheeseen, mutta eri parametreilla. Näitä *liittojakaumia* (engl. conjugate distribution) [GCSR04, s. 40][RC04, s. 30] käyttämällä posteriorijakauman laskeminen muuttuu helpommaksi, sillä uuden jakauman selvittämisen sijaan algoritmin tarvitsee päivittää vain priorijakauman parametreja. Liittojakaumien käyttäminen mallissa tosin edellyttää, että ne kuvaavat mallinnettavaa ilmiötä hyvin – pelkkä laskentahyöty ei välttämättä riitä perusteluksi. Liittojakauman käyttäminen on erityisen soveliasta silloin, kun kyseisestä mallin riippuvuudesta ei ole erityistä asiantuntijatietoa.

Mallien parametrien estimoinnin ja erityisesti koneoppimisen perusongelma on *ylisovittaminen* (engl. overfitting), jossa oppimisalgoritmi säätää mallin parametrit ku-

vaamaan tarkasti datan käyttäytymistä. Vaarana on, että algoritmi huomioikin annetun datan erityispiirteet, eikä tutkittavan ilmiön yleispiirteitä. Näin käy varsinkin silloin, kun mallissa on runsaasti (jopa tuhansia) muuttujia. Ylisovitetun mallin ennustustulokset voivat olla pahasti virheellisiä. Hierarkkisten mallien suurin etu mallintamismielessä on, että ne eivät juuri ylisovitu [GCSR04, s. 117]. Tämä johtuu siitä, että osalle muuttujista asetetaan yhteinen hyperparametri, jonka välityksellä muuttujat riippuvat toisistaan. Silloin muuttujien arvot pyrkivät kuvaamaan datan yleisiä piirteitä sen erityispiirteiden sijaan, mikä vähentää ylisovittamisen vaaraa.

Esimerkki 4.4 (Hierarkkinen malli)**.**
Kuvassa 4.5 on esitetty bayesiläinen hierarkkinen malli $\mathcal{M}$ vahingoittuneiden binääristen kuvien korjaamiseen. Vektori $\boldsymbol{y}$ sisältää datana annetut arvot vahingoittuneen kuvan binäärisille muuttujille. Vahingoittuneen kuvan yksittäisten pikselien arvojen uskotaan vastaavan todellisen kuvan pikselien arvoja $\boldsymbol{x}$ pikselikohtaisilla tarkkuuksilla $\boldsymbol{t}$. Kun 20 prosenttia vahingoittuneen kuvan pikseleistä oletetaan olevan virheellisiä, niin tarkkuus on vakio: 80 prosenttia kaikilla pikseleillä. On luontevaa olettaa muuttujien $\boldsymbol{y}$ ehdollinen riippumattomuus: kun todellisen kuvan pikselin i arvo x_i on annettu, niin vahingoittuneen kuvan pikselin i arvo y_i on riippumaton todellisen kuvan muista pikseleistä. Oletetaan hetkeksi, että todellisen kuvan pikselien arvot ovat riippumattomia toisistaan (mallia täydennetään vielä myöhemmin). Satunnaismuuttujien riippuvuudet ovat seuraavat:

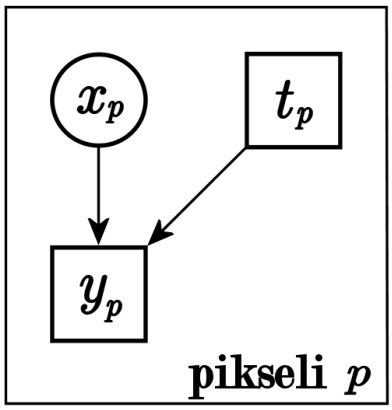
$$
\begin{aligned}
[t_i] &= 0{,}80, \\
[x_i] &\sim \text{Bernoulli}(0{,}5), \\
[y_i \mid x_i,\ t_i] &\sim \text{Bernoulli}(p_i), \text{ jossa } p_i = \begin{cases} t_i & , \text{jos } x_i = 1 \\ 1 - t_i & , \text{jos } x_i = 0. \end{cases}
\end{aligned}
$$

Kuvankorjausmallin $\mathcal{M}$ kaikkien solmujen $\mathcal{V}$ yhteisjakauma on

$$
\begin{aligned}
\Pr(\mathcal{V}) &= \Pr(\boldsymbol{x}) \cdot \overbrace{\Pr(\boldsymbol{t})}^{\text{vakio } 1} \cdot \Pr(\boldsymbol{y} \mid \boldsymbol{x},\ \boldsymbol{t}) \\
&= \Pr(\boldsymbol{x}) \cdot \Pr(\boldsymbol{y} \mid \boldsymbol{x},\ \boldsymbol{t}) \\
&= \prod_i \Pr(x_i) \cdot \Pr(y_i \mid x_i,\ t_i).
\end{aligned}
$$

4.3 Mallintaminen Markovin satunnaiskentillä

Markovin satunnaiskentän ehkä merkittävin ominaisuus on, että se muodostaa todennäköisyysjakauman satunnaiskentän solmujen eli satunnaismuuttujien arvoille.



Kuva 4.5: Hierarkkinen bayesiläinen malli vahingoittuneiden binääristen kuvien korjaamiseen. Eri pikselien välisiä riippuvuuksia ei oteta huomioon. Hierarkkisissa malleissa on tapana kuvata neliöillä kaikki vakiot ja ne muuttujat, joiden arvot annetaan datana; kaikki muut muuttujat kuvataan ympyröillä. Lisäksi käsitteelliset kokonaisuudet ympäröidään suurilla suorakulmioilla.

Sen vuoksi Markovin satunnaiskenttiä on mahdollista hyödyntää todennäköisyyslaskentaan perustuvassa mallintamisessa.

Markovin satunnaiskenttään kuuluvat muuttujat voi liittää hierarkkiseen malliin niin, että näiden muuttujien priorijakaumiksi asetetaan Markovin satunnaiskentän antamat naapurisolmujen arvoilla ehdollistetut jakaumat. Satunnaiskenttään kuuluvien muuttujien priorijakaumissa otetaan siis huomioon kaikki muut solmut, mutta vain naapurisolmujen arvojen välityksellä. Malli muuttuu tämän jälkeen sykliseksi, sillä Markovin satunnaiskentän vaikutukset ovat nimenomaan syklisiä.

Markovin satunnaiskentän muuttujiin voi liittää myös etukäteistietoa niiden mahdollisista arvoista. Esimerkissä 3.9 esiintyneiden vakioiden α_i avulla on nimittäin mahdollista kasvattaa tai vähentää yhden kokoisten klikkien potentiaalifunktioiden arvoja eli todennäköisyyksiä. Esimerkiksi jos paikkatietoaineiston jonkin datan tiedetään korreloivan tutkittavien arvojen kanssa, niin tätä dataa voi käyttää α-vakioiden laskemisessa.

Oletetaan että esimerkissä 3.9 esiintyvät vakiot $\beta_{i,j}$ saavat kaikki saman arvon β. Ising-malli sallii naapuririippuvuuden voimakkuuden β säätämisen ennen simulointia. Itsekorrelaation voimakkuuteen liittyy kuitenkin usein epävarmuutta, joka olisi hyvä ottaa huomioon mallissa. Jos vuorovaikutuksen voimakkuuden β halutaan ole-

van vakion asemesta muuttuja, sen käsittelyyn täytyy kiinnittää ylimääräistä huomiota. Tätä käsitellään vielä luvussa 5.7.

Esimerkki 4.6 (Hierarkkinen malli Markovin satunnaiskentällä)**.**
Muutetaan esimerkin 4.4 kuvankorjausmallia sillä tavalla, että todellisen kuvan pikseleiden arvoja $\boldsymbol{x}$ ei enää oleteta riippumattomiksi. Uusi malli on esitetty kuvassa 4.7. Todellisen kuvan pikseleitä vastaavat muuttujat $\boldsymbol{x}$ kuuluvat Markovin satunnaiskenttään, joka jäljittelee Ising-mallia: lähipikselien arvot muistuttavat toisiaan. Kun kuvan pikseleillä on enintään neljä naapuria, niin kuva sisältää vain yhden ja kahden pikselin kokoisia klikkejä. Kahden naapuripikselin välisen riippuvuuden voimakkuus on vakio β kaikilla kahden kokoisilla klikeillä. Markovin satunnaiskentän todennäköisyyskuvaus on sama kuin Ising-mallilla esimerkissä 3.9, mutta $\alpha_i = 0$ kaikilla pikseleillä i ja lisäksi vakio β on yhteinen. Kun $\mathcal{X}$ on kaikkien pikselimuuttujien joukko, niin ehdolliset todennäköisyydet ovat silloin muotoa

$$[x_i \mid \beta,\ \mathcal{X} \setminus \{x_i\}] \sim [x_i \mid \beta,\ \mathcal{N}(x_i)] \sim \text{Bernoulli}(p_i),$$

$$\textit{jossa } \text{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \beta \cdot \sum_{x_j \in \mathcal{N}(x_i)} (2 \cdot x_j - 1).$$

Ehdollinen todennäköisyys p_i ja sen logit-muunnos $\text{logit}(p_i)$ *on johdettu liitteessä 1. Käytetty logit-muunnos kuvaa, kuinka paljon todennäköisempi arvo 1 on arvoon 0 verrattuna. Lasketaan seuraavaksi todennäköisyys p_i sille, että kuvassa 4.8 oleva tuntematon arvo olisi ykkönen:*

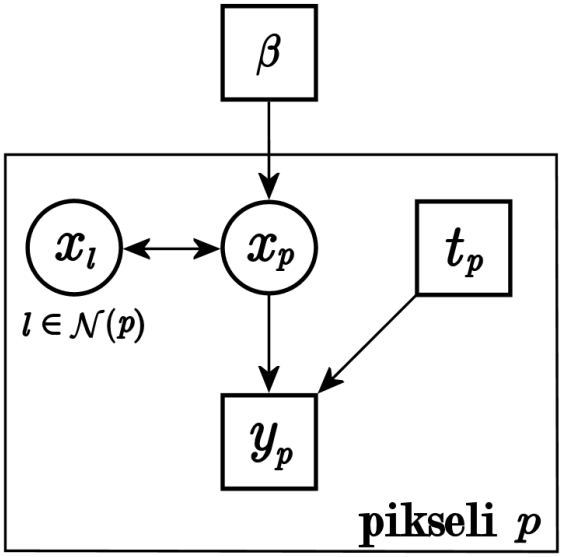
$$\text{logit}(p_i) = \beta \cdot (1 + 1 + 1 - 1) = 2 \cdot \beta \iff p_i = \frac{\exp(2 \cdot \beta)}{1 + \exp(2 \cdot \beta)}.$$

Esimerkiksi vakion β arvoilla 0,1*,* 0,5*,* 1,0 *ja* 2,0 *saadaan ykkösen todennäköisyyksiksi* 0,525*;* 0,731*;* 0,881 *ja* 0,982*.*

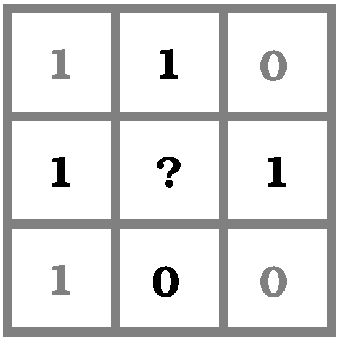
Pikselit eivät ole enää riippumattomia, joten kuvankorjausmallin $\mathcal{M}$ kaikkien solmujen $\mathcal{V}$ yhteisjakaumaksi saadaan nyt

$$\begin{aligned}
\text{Pr}(\mathcal{V}) &= \overbrace{\text{Pr}(\beta)}^{\textit{vakio 1}} \cdot \text{Pr}(\boldsymbol{x} \mid \beta) \cdot \overbrace{\text{Pr}(\boldsymbol{t})}^{\textit{vakio 1}} \cdot \text{Pr}(\boldsymbol{y} \mid \boldsymbol{x},\ \boldsymbol{t}) \\
&= \text{Pr}(\boldsymbol{x} \mid \beta) \cdot \text{Pr}(\boldsymbol{y} \mid \boldsymbol{x},\ \boldsymbol{t}) \\
&= \text{Pr}(\boldsymbol{x} \mid \beta) \cdot \prod_i \text{Pr}(y_i \mid x_i,\ t_i),
\end{aligned}$$

jossa $\text{Pr}(\boldsymbol{x} \mid \beta)$ on mallin Markovin satunnaiskentän todennäköisyyskuvaus. Tätä kuvankorjausmallia sovelletaan myöhemmin esimerkissä 5.19.



Kuva 4.7: Hierarkkinen bayesiläinen malli vahingoittuneiden binääristen kuvien korjaamiseen. Naapuripikselien väliset riippuvuudet otetaan huomioon. Joukko $\mathcal{N}(p)$ sisältää pikselin x_p naapurit.



Kuva 4.8: Markovin satunnaiskentän asetelma, jossa yhden pikselin arvoille lasketaan todennäköisyydet.

5 Bayesiläisten mallien laskennalliset menetelmät

Seuraavaksi näytetään, millä tavalla bayesiläisiä malleja on mahdollista käsitellä laskennallisesti. Lopullisena päämääränä on muodostaa approksimaatio bayesiläisen mallin posteriorijakaumalle, kun joidenkin muuttujien arvot annetaan datana. Monte Carlo -menetelmän ja Markovin ketjujen esittelyn jälkeen tarkastellaan MCMC-menetelmää. Luvun lopussa käydään vielä läpi MCMC-menetelmän toteuttamisen ja analysoinnin yksityiskohtia sekä soveltamista bayesiläisiin Markovin satunnaiskenttiä hyödyntäviin malleihin.

5.1 MAP-estimaatti

Bayesiläisen mallin posteriorijakauman analyyttinen johtaminen on mahdollista vain hyvin yksinkertaisissa tilanteissa. Ennen kuin MCMC-menetelmät yleistyivät, jouduttiin käyttämään muita menetelmiä, joilla pystyttiin analysoimaan bayesiläisiä malleja laskennallisesti tehokkaasti.

Yksi menetelmä perustuu siihen, että koko posteriorijakauman sijaan selvitetään vain posteriorijakauman kaikkein todennäköisin mallin muuttujien arvoyhdistelmä. Tällä on yhteyksiä suurimman uskottavuuden estimaatin kanssa, mutta uskottavuustiheyden $\textsc{pr}(\mathcal{D} \mid \boldsymbol{\theta})$ sijaan maksimoidaankin posterioritodennäköisyystiheyttä. Tällaista posteriorijakauman maksimia kutsutaan *MAP-estimaatiksi* (engl. maximum a posteriori) [Hec97], ja se antaa mallin muuttujille sellaiset kiinteät arvot, että vastaava posterioritodennäköisyystiheys on suurimmillaan. MAP-estimaatti on siis

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \ \textsc{pr}(\boldsymbol{\theta} \mid \mathcal{D}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \ \textsc{pr}(\boldsymbol{\theta}) \cdot \textsc{pr}(\mathcal{D} \mid \boldsymbol{\theta}).$$

Tässä jätetään Bayesin kaavan (4.2) jakaja huomiotta, sillä posteriorijakauman todennäköisyystiheydet ovat verrannollisia keskenään. Monesti jo yksi arvoyhdistelmä (MAP-estimaatti) riittää kuvaamaan mallin muuttujien arvoja riittävän hyvin. Ongelmana on, että satunnaismuuttujiin sisältyy usein epävarmuutta, eikä ole välttämättä järkevää kiinnittää arvoja MAP-estimaatin mukaisiksi.

MAP-estimaatti saadaan selvitettyä posteriorijakaumasta paikallisen etsinnän menetelmillä ilman koko posteriorijakauman selvittämistä. Ehtona on, että posteriorijakaumasta voidaan laskea toisiinsa verrannollisia todennäköisyystiheyksiä annetuille arvovektoreille. *Paikallisessa etsinnässä* (engl. local search) etsintäalgoritmi

lähtee liikkeelle jostakin satunnaisesta arvovektorista ja pyrkii löytämään todennäköisimmän arvovektorin pienten muutosten kautta. Maksimin löytäminen on varmaa vain erikoistapauksissa, joissa algoritmin annetaan etsiä maksimia riittävän pitkään. Käytännössä paikallisen etsinnän algoritmit löytävät hyviä maksimiarvoja varsin nopeasti. Esimerkiksi simuloitu jäähdytys (engl. simulated annealing) [KGV83] on suosittu keino löytää funktion (tässä posteriorijakauman tiheysfunktion) globaali maksimikohta. Simuloidun jäähdytyksen sijaan voidaan käyttää geneettisiä algoritmeja (engl. genetic algorithms) [SP94], joiden toiminta perustuu arvovektorin evoluutioon. Myös muunnellun *EM-algoritmin* [RC04, luku 5.3.2][DLR77] käyttäminen on mahdollista.

5.2 Monte Carlo -menetelmät

Monte Carlo -menetelmillä tarkoitetaan yleisesti satunnaisuuteen perustuvia algoritmeja. Tarkastellaan integrointiongelmaa, jossa halutaan selvittää todennäköisyysjakauman π mukaan jakautuneesta satunnaismuuttujasta X kuvaajan g määrittämän ominaisuuden odotusarvo. Olkoon $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ todennäköisyysjakaumasta π poimittu satunnainen n:n kokoinen otos. Kun n on riittävän suuri, niin suurten lukujen lain perusteella pätee

$$\mathtt{E}[g(X)] = \int_{\boldsymbol{s}} g(\boldsymbol{s}) \cdot \pi(\boldsymbol{s}) \; d\boldsymbol{s} \approx \frac{1}{n} \cdot \sum_{i=1}^{n} g(\boldsymbol{x}^{(i)}). \tag{5.1}$$

Edellä olevaa approksimaatiota (5.1) kutsutaan *Monte Carlo -integroinniksi* [RC04, luku 3.2]. Suurten lukujen lain mukaan approksimaatio on sitä tarkempi, mitä suurempi satunnaisotos on käytettävissä. Täysin riippumattomien satunnaisvektorien poimiminen on monesti varsin hankalaa. Annettu tulos pätee kuitenkin myös tapauksissa, jossa satunnaisvektorit korreloivat keskenään. Ehtona on, että satunnaisotoksen sisältämät vektorit esiintyvät otoksessa samassa suhteessa kuin jakaumassa π. Monte Carlo -integrointi on usein ainoa keino approksimoida posteriorijakauman kiinnostavien muuttujien reunajakaumia.

5.3 Satunnaisvektorien poiminta perinteisillä tavoilla

Monte Carlo -integroinnin käyttäminen vaatii, että todennäköisyysjakaumasta π on mahdollista poimia satunnaisvektoreita. Valitettavasti satunnaisvektorien poimiminen todennäköisyysjakaumasta on helppoa vain harvoissa tapauksissa.

Halutusta jakaumasta on helppo poimia satunnaisvektoreita, jos jakauman kertymäfunktion käänteisfunktio on selvitettävissä [RC04, luku 2.1.2]. Tämä kertymäfunktion käänteisfunktioon perustuva menetelmä on käytännössä mahdollinen vain kaikkein yksinkertaisimmissa tilanteissa. Kertymäfunktion käänteisfunktion laskeminen saattaa nimittäin olla laskennallisesti hyvin hankalaa ja joskus sellaista ei edes voi lausua suljetussa muodossa.

Oletetaan että halutaan tuottaa yksiulotteisen todennäköisyysjakauman π mukaan jakautuneita satunnaisarvoja. *Hylkäyspoiminnassa* (engl. rejection sampling) [RC04, luku 2.3.2] on ideana, että tuotetaan ensin helpomman jakauman q satunnaisarvoja ja hylätään näistä sitten osa. Hylkäyksiä jatketaan niin kauan, kunnes jakaumasta q tuotettu satunnaisarvo hyväksytään jakauman π satunnaisarvoksi. Jakaumaa q kutsutaan jakauman π *ehdotusjakaumaksi* (engl. proposal distribution). Tehokkuuden kannalta paras tilanne on, jos ehdotusjakaumasta q on helppo poimia satunnaisarvoja ja jakauman q muoto muistuttaa kiinnostavan jakauman π muotoa. Valitaan ehdotusjakauma q ja jokin vakio c, joille pätee $\pi(x) \leqslant c \cdot q(x)$ kaikilla muuttujan x arvoilla. Vakiolla c kerrottu ehdotusjakauma q ei siis saa koskaan pienempiä arvoja kuin jakauma π. Vakio c tulisi pitää tehokkuussyistä mahdollisimman pienenä. Algoritmin tehokkuus riippuu siitä, kuinka helposti ehdotusjakaumasta q voidaan poimia satunnaisarvoja ja kuinka usein niitä joudutaan hylkäämään. Hylkäyspoiminnan toiminta on esitetty algoritmissa 5.2 ja havainnollistettu kuvassa 5.3.

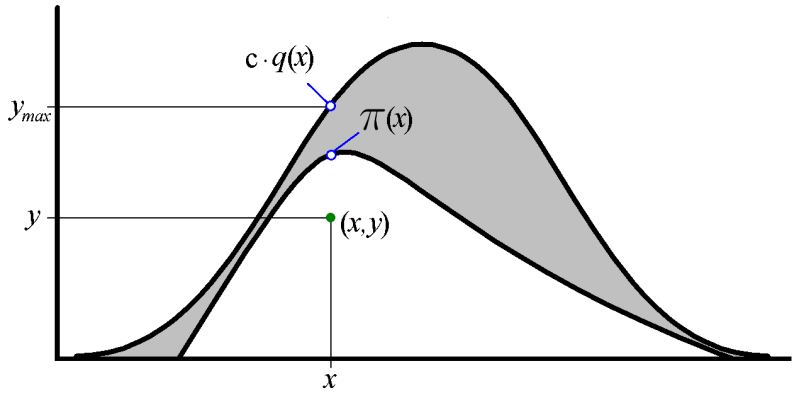
Algoritmi 5.2 Hylkäyspoiminta

Syöte: yksiulotteiset todennäköisyyskuvaukset π ja q sekä kerroin $c > 0$

Tulos: satunnaisarvo x todennäköisyysjakaumasta π

1: **toista**
2: $x \leftarrow$ SATUNNAINEN(q) ▷ poimitaan satunnaisarvo ehdotusjakaumasta q
3: $y_{max} \leftarrow c \cdot q(x)$ ▷ skaalatun ehdotusjakauman $c \cdot q$ arvo kohdassa x
4: $y \leftarrow$ SATUNNAINEN(Tas(0, y_{max}))
5: **kunnes** $y \leqslant \pi(x)$ ▷ jääkö koordinaattipiste (x, y) jakauman π sisälle?
6: **palauta** x

Käydään seuraavaksi läpi hylkäyspoiminta-algoritmin aikavaativuus. Oletetaan että algoritmin hyväksymistodennäköisyys on p. Olkoon satunnaismuuttuja Y yritysten lukumäärä, kunnes hyväksyminen tapahtuu. Silloin $\mathtt{E}[Y] = \sum_{i=1}^{\infty} i \cdot (1-p)^{i-1} \cdot p = \frac{1}{p}$. Jos ehdotusjakaumasta q voidaan poimia satunnaisarvoja vakioajassa, niin kiinteällä hyväksymistodennäköisyydellä p myös hylkäyspoiminta-algoritmin odotusarvoinen



Kuva 5.3: Hylkäyspoiminta-algoritmin toiminta yksiulotteisella jatkuvalla todennäköisyyskuvauksella π ja ehdotusjakaumalla q. Ehdotusjakaumasta q poimitaan satunnaisarvo x, joka tulkitaan x-akselin arvoksi. Sen jälkeen arvotaan satunnaisluku y tasaiselta väliltä $[0, y_{max}]$ ja tarkistetaan, jos koordinaatiston piste (x, y) olisi jakauman π sisällä. Kuvassa oleva piste (x, y) on tässä tapauksessa jakauman π sisällä ja satunnaisarvo x hyväksytään; jos piste olisikin harmaalla alueella, niin satunnaisarvo x hylättäisiin.

aikavaativuus on vakio.

Useamman kuin yhden ulottuvuuden todennäköisyysjakaumista poimitaankin satunnaisarvojen sijaan satunnaisvektoreita, joissa on yksi arvo jokaiselle ulottuvuudelle. Teoriassa aiemmin esitetyt satunnaisarvojen poimintamenetelmät yleistyvät kaikki myös korkeampiin ulottuvuuksiin, mutta ne toimivat käytännössä vain yksinkertaisilla, esimerkiksi yksiulotteisilla, jakaumilla. Syynä on, että kun todennäköisyysjakauman ulottuvuuksien (mallin muuttujien) määrä kasvaa useisiin tuhansiin, niin sopivaa ehdotusjakaumaa on lähes mahdotonta löytää [Nea93, luku 3.2]. Yksiulotteisilla jakaumilla hyvin toimineet ehdotusjakaumat eivät yleensä toimi suurissa ulottuvuuksissa, sillä hylkäyspoiminnan hyväksymistodennäköisyys vähenee eksponentiaalisesti ulottuvuuksien määrän mukaan.

5.4 Satunnaisvektorien poiminta Markovin ketjun avulla

Aiemmin todettiin, että satunnaisvektorien poimiminen moniulotteisesta todennäköisyysjakaumasta on hankalaa perinteisin menetelmin. Ratkaisuna tähän on kehitetty Markovin ketjuja simuloivia algoritmeja, jotka poimivat satunnaisvektoreita

tehokkaasti myös moniulotteisista jakaumista. Seuraavaksi käydään ensin läpi Markovin ketjujen perusteet ja esitellään sitten tärkeimmät Markovin ketjuja simuloivat poiminta-algoritmit: Metropolis–Hastings- ja Gibbsin algoritmit. Lopuksi annetaan lyhyet kuvaukset muista vaihtoehtoisista algoritmeista.

5.4.1 Markovin ketjut

Tässä aliluvussa käsitellään Markovin ketjujen perusteet.

Määritelmä 5.4 (Markovin ketju [Gam97, luku 4])**.**
Olkoon $(X^{(0)},\ X^{(1)},\ X^{(2)},\ \ldots)$ *ääretön jono (toisistaan riippuvia) satunnaismuuttujia, joilla on kaikilla sama arvoalue* $\Omega \subseteq \mathbb{R}^m$ *jollakin positiivisella kokonaisluvulla* m*. Olkoon* PR *näiden satunnaismuuttujien yhteisjakauma. Jonoa kutsutaan* Markovin ketjuksi *(engl. Markov chain), jos kaikilla* $t \in \mathbb{N}$ *pätee*

$$\begin{aligned}&\mathrm{PR}(X^{(t+1)} \mid X^{(0)},\ X^{(1)},\ \ldots,\ X^{(t)})\\ =&\mathrm{PR}(X^{(t+1)} \mid X^{(t)}).\end{aligned}$$

Määritelmä 5.4 kuvaa satunnaisprosessia. Markovin ketjun yhteisjakauma määrää todennäköisyystiheydet kaikille jonoille, jossa satunnaismuuttujille on asetettu kiinteät arvot. Myös tällaisia kiinteiden arvojen jonoja on tavallista kutsua Markovin ketjuksi (vaikka ne eivät olekaan enää satunnaisia), ja niin tehdään tässäkin tutkielmassa.

Määritelmässä 5.4 Markovin ketjun satunnaismuuttujien arvojen todennäköisyystiheydet riippuvat jonon aiemmista arvoista, mutta vain juuri edeltävän arvon välityksellä. Edellisen arvon tunteminen vastaa siis koko ketjun historian tuntemista. Aiemmin esitetty Markovin satunnaiskenttä on itse asiassa eräs Markovin ketjun yleistys.

Rinnastetaan jatkossa Markovin ketjun satunnaismuuttujien arvoalueen Ω arvot ja ketjun *tilat*. Voidaan ajatella, että Markovin ketju on jokaisella hetkellä $t \in \mathbb{N}$ jossakin tilassa (eli saa arvon) $X^{(t)}$. Markovin ketjun satunnaismuuttujan $X^{(t+1)}$ arvo kuvaa tilaa, johon ketju siirtyy seuraavaksi. Ketjussa eteneminen voidaan ajatella tilojen välisinä siirtyminä.

Markovin ketju voidaan ilmaista yksikäsitteisesti *alkutilakuvauksen* ja *siirtymäkuvausten* (engl. transition kernel) sisältämien todennäköisyystiheyksien avulla [Nea93,

luku 3.3]. Alkutilakuvaus määrittää todennäköisyysjakauman ketjun ensimmäiselle tilalle $X^{(0)}$ ja siirtymäkuvaukset sitä seuraaville siirtymille.

Alkutilakuvaus h on todennäköisyyskuvaus tila-avaruudelle Ω, ja se määrittää ensimmäisen satunnaismuuttujan $X^{(0)}$ kaikille arvoille $j \in \Omega$ todennäköisyystiheydet

$$h(j) = \text{PR}(X^{(0)} = j).$$

Jos tila-avaruus Ω on diskreetti, niin alkutilakuvaus voidaan ilmaista todennäköisyysvektorilla $\boldsymbol{h}$, joka sisältää todennäköisyyden jokaista tilaa kohti.

Hetken $t \in \mathbb{N}$ siirtymäkuvauksen $s^{(t)}$ arvot ovat todennäköisyystiheyksiä

$$s^{(t)}(i, j) = \text{PR}(X^{(t+1)} = j \mid X^{(t)} = i),$$

joiden tulkinta on seuraava: jos Markovin ketju olisi hetkellä t tilassa i, niin millä todennäköisyystiheydellä se siirtyisi tilaan j hetkeksi $t + 1$? Jokainen $s^{(t)}(i, \cdot)$ on todennäköisyyskuvaus tila-avaruudelle Ω. Diskreetin tila-avaruuden Ω tapauksessa siirtymäkuvaus ilmaistaan usein hetken t *siirtymämatriisina* $\boldsymbol{S}^{(t)}$, jonka rivin i ja sarakkeen j arvo $S_{i,j}^{(t)}$ kertoo todennäköisyyden siirtyä hetkellä t tilasta i tilaan j.

Usein asetetaan yksinkertaisuuden vuoksi $s^{(t)} = s$ kaikilla t. Silloin Markovin ketju on *aikahomogeeninen* eli sen siirtymäkuvaus pysyy samana jokaisena hetkenä t. Erityisesti diskreetin tila-avaruuden Ω tapauksessa siirtymämatriiseille pätee silloin $\boldsymbol{S}^{(t)} = \boldsymbol{S}$ kaikilla $t \in \mathbb{N}$. Tästä eteenpäin oletetaan, että kaikki tarkkailtavat Markovin ketjut ovat aikahomogeenisia.

Esimerkki 5.5 (Sään ennustaminen Markovin ketjulla)**.**
Käydään esimerkkinä läpi Markovin ketju, joka ennustaa säätilaa. Kyseinen leikkimalli ottaa huomioon nykyisen säätilan ja ennustaa sen avulla säätilan tunnin päähän. Yksi siirtymä Markovin ketjussa vastaa siis yhden tunnin siirtymää tosiajassa.

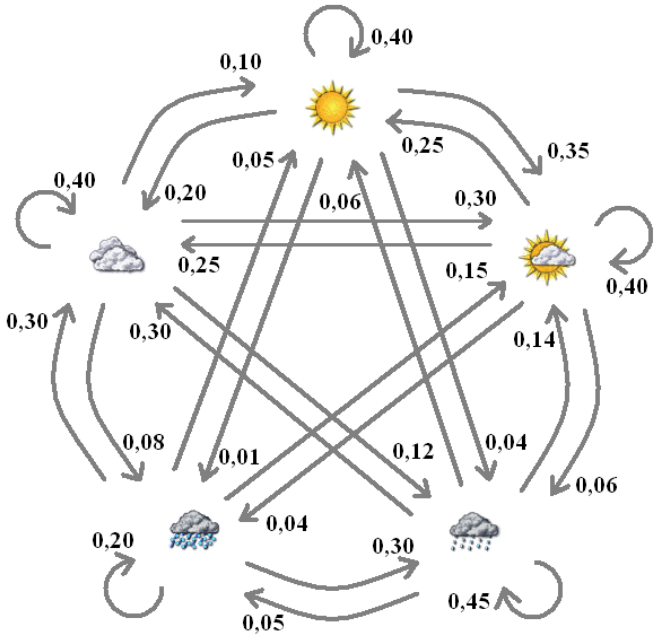
Ketjun tila-avaruus on diskreetti joukko $\Omega = \{AU,\ PU,\ PI,\ SA,\ RA\}$*, jossa tilat ovat AUrinkoinen, PUolipilvinen, PIlvinen, SAteinen ja RAnkkasateinen. Yön aikana selkeä sää tulkitaan aurinkoiseksi. Määritellään ketjulle aikahomogeeninen siirtymämatriisi*

$$\boldsymbol{S} = \begin{bmatrix} 0{,}40 & 0{,}35 & 0{,}20 & 0{,}04 & 0{,}01 \\ 0{,}25 & 0{,}40 & 0{,}25 & 0{,}06 & 0{,}04 \\ 0{,}10 & 0{,}30 & 0{,}40 & 0{,}12 & 0{,}08 \\ 0{,}06 & 0{,}14 & 0{,}30 & 0{,}45 & 0{,}05 \\ 0{,}05 & 0{,}15 & 0{,}30 & 0{,}30 & 0{,}20 \end{bmatrix}.$$

Mallin siirtymämatriisin todennäköisyydet on hahmoteltu myös kuvaan 5.6. Siirtymämatriisin rivit ja sarakkeet vastaavat järjestyksessä tiloja AU, PU, PI, SA ja RA. Siirtymämatriisin jokainen rivi määrittää todennäköisyysjakauman ketjun seuraavalle tilalle. Diagonaali sisältää todennäköisyydet tilojen siirtymille takaisin itseensä.

Markovin ketjun alku (aloitustila AUrinkoinen) voisi esimerkiksi olla:

$$(AU,\ AU,\ PU,\ AU,\ PI,\ SA,\ SA,\ SA,\ \ldots).$$



Kuva 5.6: Siirtymien todennäköisyydet Markovin ketjuihin perustuvan säämallin tilojen välillä. Tilat ylhäältä myötäpäivään lukien: AU, PU, SA, RA ja PI.

Siirtymäkuvaus s asettaa todennäköisyystiheydet välittömästi seuraaville tiloille, kun nykyinen tila tunnetaan. Määritellään n askeleen yhdistetty siirtymäkuvaus jostakin hetkestä $t \in \mathbb{N}$ alkaen seuraavasti:

$$s^n(i,j) = \text{PR}(X^{(t+n)} = j \mid X^{(t)} = i).$$

Jos ketju on tilassa i hetkellä t, niin tämä siirtymäkuvaus antaa todennäköisyystiheyden päätyä tilaan j yhteensä n siirtymän jälkeen. Diskreetin tila-avaruuden Ω aikahomogeenisen siirtymämatriisin n:s potenssi $\boldsymbol{S}^n$ sisältää nämä samat n askeleen päähän ulottuvat todennäköisyydet.

Joissakin Markovin ketjuissa on havaittavissa, että kun ketjun tiloja tutkitaan kyllin pitkälle, niin tilojen esiintymistiheydet lähestyvät kohti joitakin kiinteitä arvoja.

Määritelmä 5.7 (Markovin ketjun tasapainojakauma [Gam97, luku 4.7.2])**.** *Olkoon s Markovin ketjun siirtymäkuvaus ja π todennäköisyysjakauma, joka kertoo todennäköisyystiheydet Markovin ketjun jokaiselle tilalle. Jos pätee*

$$\pi(j) = \int_{i \in \Omega} \pi(i) \cdot s(i,j)\ di$$

kaikilla tiloilla $j \in \Omega$, niin jakaumaa π kutsutaan Markovin ketjun tasapainojakaumaksi *(engl. stationary, equilibrium, invariant distribution). Jos tila-avaruus Ω on diskreetti, niin integraali tulkitaan summaksi. Diskreetille tapaukselle on myös vaihtoehtoinen merkintätapa: todennäköisyysvektori $\boldsymbol{\pi}$ on ketjun tasapainojakauma, jos sen siirtymämatriisille $\boldsymbol{S}$ pätee*

$$\boldsymbol{\pi} = \boldsymbol{\pi} \cdot \boldsymbol{S}.$$

Kaikilla Markovin ketjuilla ei ole tasapainojakaumaa ja joillakin on jopa useita. Seuraavaksi tarkastellaan, milloin yksikäsitteinen tasapainojakauma on olemassa ja milloin ketju konvergoituu kohti sellaista.

Diskreetin tila-avaruuden Markovin ketju on *jaksoton* (engl. aperiodic), jos joukon $\{n \geqslant 1 \mid S^n_{j,j} > 0\}$ eli syklien pituuksien joukon suurin yhteinen tekijä on 1 jokaisella tilalla $j \in \Omega$ erikseen. Ketjun jaksollisuus tarkoittaa, että siinä on säännöllisin välein hetkiä, jolloin joidenkin tilojen esiintymisen todennäköisyys on nolla. Kun Markovin ketjun tila-avaruus on jatkuva, niin tarkastellaankin joukon $\{n \geqslant 1 \mid s^n(J,J) > 0\}$ suurinta yhteistä tekijää kaikilla Borel-joukoilla $J \subseteq \Omega$. Borel-joukkojen esittely on luonteeltaan turhan tekninen tähän esitykseen; riittää tietää, että ”tavallisimmat” osajoukot ovat Borel-joukkoja.

Muodostetaan koko tila-avaruuden Ω kattava suunnattu verkko, jossa solmujen (tilojen) i ja j välillä on suunnattu kaari, jos tilasta i on aidosti positiivinen todennäköisyystiheys siirtyä suoraan tilaan j eli $s(i,j) > 0$. Jos syntynyt verkko on vahvasti yhtenäinen, niin vastaava Markovin ketju on *yhtenäinen* (engl. irreducible). Tämä tarkoittaa diskreetissä tila-avaruudessa, että jokaisesta tilasta i on aidosti positiivinen todennäköisyys päästä mihin tahansa tilaan j ainakin jollakin määrällä siirtymiä eli $\max\{S^n_{i,j} \mid n \geqslant 1\} > 0$ kaikilla $i,j \in \Omega$. Ketjun ensimmäinen tila $X^{(0)}$ ei siis voi rajoittaa myöhemmin saavutettavia tiloja, joten aloitustilan merkitys katoaa ajan myötä. Jatkuvassa tila-avaruudessa tarkastellaan ehtoa $\max\{s^n(i,J) \mid n \geqslant 1\} > 0$ kaikilla tiloilla $i \in \Omega$ ja Borel-joukoilla $J \subseteq \Omega$.

Oletetaan että Markovin ketju on jaksoton ja yhtenäinen. Jos ketjulla on tasapainojakauma π, niin π on ketjun yksikäsitteinen tasapainojakauma ja ketju konvergoituu sitä kohti. Toisin sanoen

$$\lim_{n\to\infty} s^n(i,j) = \pi(j)$$

kaikilla tiloilla $j \in \Omega$ ja riippumatta aloitustilasta $i \in \Omega$ [Tie96][Gam97, luku 4.7]. Tiloille jaettu todennäköisyysmassa jakaantuu lopulta tasapainojakauman mukaisesti, riippumatta alkutilasta. Kun tasapainojakauma on kerran saavutettu (konvergoituminen), niin ketjun jatkaminen ei enää muuta tilojen asymptoottisia esiintymistiheyksiä.

Määritelmä 5.8 (Markovin ketjun kääntyvyysehto [Gam97, luku 4.7.2])**.**
Markovin ketjun siirtymäkuvaus s ja todennäköisyysjakauma π toteuttavat kääntyvyysehdon *(engl. reversibility, detailed balance), jos kaikille tiloille $i, j \in \Omega$ pätee*

$$\pi(i) \cdot s(i,j) = \pi(j) \cdot s(j,i).$$

Lause 5.9.
Jos Markovin ketjun siirtymäkuvaus s ja todennäköisyysjakauma π toteuttavat kääntyvyysehdon, niin silloin π on Markovin ketjun tasapainojakauma.

Todistus. Tarkastellaan mielivaltaista Markovin ketjun tilojen Borel-osajoukkoa $J \subseteq \Omega$. Ensinnäkin pätee

$$\int_{i\in\Omega} \pi(i) \cdot s(i,J)\ di = \int_{i\in\Omega} \int_{j\in J} \pi(i) \cdot s(i,j)\ dj\ di = (*).$$

Kääntyvyysehdon mukaan $\pi(i) \cdot s(i,j) = \pi(j) \cdot s(j,i)$, joten saadaan

$$(*) = \int_{i\in\Omega} \int_{j\in J} \pi(j) \cdot s(j,i)\ dj\ di.$$

Integroitavat termit ovat ei-negatiivisia, joten Fubinin lauseen mukaan integrointijärjestyksen saa vaihtaa. Jatketaan seuraavasti

$$(*) = \int_{j\in J} \int_{i\in\Omega} \pi(j) \cdot s(j,i)\ di\ dj$$

$$(*) = \int_{j\in J} \pi(j) \cdot \int_{i\in\Omega} s(j,i)\ di\ dj.$$

Koska $s(j,\cdot)$ on todennäköisyysjakauma kaikilla $j \in \Omega$, niin $\int_{i\in\Omega} s(j,i)\ di = 1$, ja saadaan lopulta

$$(*) = \int_{j\in J} \pi(j)\ dj.$$

Koska tämä pätee kaikille Borel-joukoille $J \subseteq \Omega$, niin todennäköisyyskuvaus π on Markovin ketjun tasapainojakauma. □

Jos Markovin ketju on yhtenäinen, niin määritelmän 5.8 mukainen tasapainojakauma π on itse asiassa yksikäsitteinen ja Markovin ketju konvergoituu sitä kohti [Gam97, s. 107]. Kääntyvyysehto 5.8 ei ole välttämätön ehto yksikäsitteisen tasapainojakauman olemassaololle, mutta riittävä. Kääntyvyysehtoa käytetään myöhemmin osoittamaan, että tiettyjen algoritmien tuottamilla Markovin ketjuilla on tasapainojakauma, jota kohti ketjut konvergoituvat.

Esimerkki 5.10 (Säämallin tasapainojakauma)**.**
Jos esimerkissä 5.5 esitetty leikkisäämalli kuvaisi hyvin todellisen sään käyttäytymistä, niin kaikkien säätyyppien todelliset esiintymistiheydet vastaisivat ketjun tasapainojakauman arvoja. Mallin Markovin ketju on selvästi jaksoton ja yhtenäinen, sillä jokaisesta tilasta on suorat siirtymät kaikkiin tiloihin. Käytetään hyväksi diskreetin tasapainojakauman $\boldsymbol{\pi}$ *(välttämätöntä) ominaisuutta* $\sum_{i\in\Omega} \pi_i = 1$ *ja ratkaistaan sen avulla tasapainojakauma lineaarisesta yhtälöryhmästä* $\boldsymbol{\pi} = \boldsymbol{\pi} \cdot \boldsymbol{S}$*, jossa* $\boldsymbol{S}$ *on aiemmin esitetty säätilojen siirtymämatriisi. Lopputuloksena saadaan tasapainojakaumaksi*

$$\boldsymbol{\pi} \approx (0{,}197\ ;\ 0{,}309\ ;\ 0{,}294\ ;\ 0{,}143\ ;\ 0{,}056),$$

jonka perusteella puolipilvinen olisi niukasti yleisin säätyyppi. Jos matriisia $\boldsymbol{S}$ *kertoo itsellään useita kertoja, niin matriisin rivivektorien todennäköisyysarvot konvergoituvat kohti tätä tasapainojakaumaa. Kuvattu Markovin ketju ei kuitenkaan täytä kääntyvyysehtoa 5.8, sillä esimerkiksi tiloilla AU ja SA pätee*

$$\pi_{AU} \cdot S_{AU,SA} \approx 0{,}197 \cdot 0{,}04 = 0{,}00788 \neq 0{,}00858 = 0{,}143 \cdot 0{,}06 \approx \pi_{SA} \cdot S_{SA,AU}.$$

5.4.2 MCMC-menetelmä

Palataan tilanteeseen, jossa kiinnostavasta todennäköisyysjakaumasta π halutaan poimia satunnaisvektoreita. Aiemmin todettiin, että perinteiset satunnaisarvojen poiminta-algoritmit muuttuvat lähes käyttökelvottomiksi, jos jakaumassa π on runsaasti ulottuvuuksia. Myös moniulotteisista jakaumista on mahdollista poimia satunnaisvektoreita, jos poiminta-algoritmi perustuu Markovin ketjujen käyttämiseen.

Jos Markovin ketju konvergoituu kohti jotakin tasapainojakaumaa, niin sen tiloja voidaan käyttää tasapainojakaumasta poimittuina (riippuvina) satunnaisvektoreina.

MCMC-menetelmässä onkin ideana muodostaa Markovin ketju, jolla on tasapainojakaumana kiinnostava jakauma π.

Markovin ketju Monte Carlo -menetelmässä (engl. Markov chain Monte Carlo, MCMC) [Gam97, luku 4.8] simuloidaan Markovin ketjua ja sovelletaan Monte Carlo -integrointia. Monte Carlo -integroinnin tarvitsemat satunnaisvektorit saadaan Markovin ketjun simulaatiosta: ketjun tilat eri hetkillä tulkitaan tasapainojakauman satunnaisvektoreiksi. Markovin ketjun simulaation avulla on mahdollista poimia satunnaisvektoreita myös ilman, että Monte Carlo -integrointia käytetään lainkaan. Ne esiintyvät yhdessä kuitenkin niin usein, että MCMC-menetelmästä puhutaan silloinkin, kun painopiste on pelkästään satunnaisvektorien poimimisessa.

MCMC-menetelmien suurin etu on, että niiden avulla on mahdollista poimia satunnaisvektoreita myös sellaisista jakaumista, joita muut menetelmät eivät pysty käsittelemään. Myös yleiskäyttöisyys on merkittävä etu. MCMC-menetelmien pahin heikkous on, että niiden teoreettinen analyysi on erittäin vaikeata. Ei ole koskaan täysin varmaa, ovatko tuotetut satunnaisvektorit otoksia halutusta jakaumasta. Lisäksi MCMC-simulaatiot vievät runsaasti aikaa.

5.4.3 Metropolis–Hastings-poiminta-algoritmi

Jotta Markovin ketjun simulointi tuottaisi (riippuvia) satunnaisvektoreita halutusta todennäköisyysjakaumasta, niin ketjun tasapainojakauman tulee vastata tätä jakaumaa. Tarvitaan siis menetelmä, jolla voidaan tuottaa annettua tasapainojakaumaa vastaava Markovin ketju.

Metropolis–Hastings-poiminta-algoritmi [Has70][Nea93, luku 4] on yksi yleiskäyttöisimmistä menetelmistä tuottaa Markovin ketju, jolla on haluttu tasapainojakauma. Oletetaan että Markovin ketjulle on annettu (sallittu) alkutila $X^{(0)}$. Kun seuraavaksi esiteltävää algoritmia toistetaan n kertaa, ketjulle saadaan tilat $X^{(1)}, \ldots, X^{(n)}$.

Olkoon Markovin ketjun tila-avaruus Ω. Metropolis–Hastings-algoritmi vaatii, että todennäköisyysjakaumasta π voidaan laskea kahden todennäköisyystiheyden suhde $\frac{\pi(\boldsymbol{z})}{\pi(\boldsymbol{x})}$ kaikilla tiloilla $\boldsymbol{z}, \boldsymbol{x} \in \Omega$. Lisäksi jokaisella tilalla $\boldsymbol{x} \in \Omega$ on oltava *ehdotusjakauma* (engl. proposal distribution) $q(\cdot \mid \boldsymbol{x})$ tila-avaruudessa Ω. Algoritmi 5.11 tekee yhden siirtymän Markovin ketjussa.

Oletetaan että Markovin ketju on tilassa $\boldsymbol{x} \in \Omega$ hetkellä $t \in \mathbb{N}$ eli $X^{(t)} = \boldsymbol{x}$, ja että kyseinen tila on mahdollinen eli $\pi(\boldsymbol{x}) > 0$. Algoritmi tuottaa nykyiselle tilal-

Algoritmi 5.11 Metropolis–Hastings-poiminta

Syöte: Markovin ketjun tila $X^{(t)} = \boldsymbol{x}$ sekä todennäköisyyskuvaukset π ja $q(\cdot \mid \boldsymbol{x})$

Tulos: Markovin ketjun tila $X^{(t+1)}$

1: $\boldsymbol{z} \leftarrow$ SATUNNAINEN$(q(\cdot \mid \boldsymbol{x}))$ ▷ poimitaan ehdotus uudeksi tilaksi
2: $\alpha \leftarrow \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z})}{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x})}\right\}$ ▷ lasketaan hyväksymistodennäköisyys
3: $rnd \leftarrow$ SATUNNAINEN(Tas(0, 1))
4: **jos** $rnd \leqslant \alpha$ **niin**
5: **palauta** $\boldsymbol{z}$ ▷ hyväksytään ehdotus
6: **muuten**
7: **palauta** $\boldsymbol{x}$ ▷ hylätään ehdotus
8: **lopeta jos**

le $\boldsymbol{x}$ *ehdotuksen* (engl. proposal) $\boldsymbol{z}$ seuraavaksi tilaksi $X^{(t+1)}$. Ehdotus $\boldsymbol{z}$ poimitaan ehdotusjakaumasta $q(\cdot \mid \boldsymbol{x})$. Sen jälkeen algoritmi selvittää tilojen $\boldsymbol{z}$ ja $\boldsymbol{x}$ todennäköisyystiheyksien $\pi(\boldsymbol{z})$ ja $\pi(\boldsymbol{x})$ suhteen todennäköisyysjakaumassa π sekä ehdotusjakaumien todennäköisyystiheyksien $q(\boldsymbol{x} \mid \boldsymbol{z})$ ja $q(\boldsymbol{z} \mid \boldsymbol{x})$ suhteen. Sitten algoritmi laskee näiden avulla *hyväksymistodennäköisyyden* sille, että nykyisestä tilasta $\boldsymbol{x}$ siirrytään ehdotettuun tilaan $\boldsymbol{z}$:

$$\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z})}{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x})}\right\}. \tag{5.12}$$

Algoritmi asettaa Markovin ketjun seuraavaksi tilaksi $X^{(t+1)}$ ehdotetun tilan $\boldsymbol{z}$ todennäköisyydellä $\alpha(\boldsymbol{z} \mid \boldsymbol{x})$ ja nykyisen tilan $\boldsymbol{x}$ todennäköisyydellä $1 - \alpha(\boldsymbol{z} \mid \boldsymbol{x})$. Hyväksymistodennäköisyys $\alpha(\boldsymbol{z} \mid \boldsymbol{x})$ korjaa ehdotusjakauman q ja tasapainojakauman π väliset eroavaisuudet. Markovin ketjun siirtymäkuvaus s voidaan ilmaista Metropolis–Hastings-algoritmissa siis asettamalla $s(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{z} \mid \boldsymbol{x}) \cdot \alpha(\boldsymbol{z} \mid \boldsymbol{x})$, kun $\boldsymbol{x}, \boldsymbol{z} \in \Omega$ ja $\boldsymbol{x} \neq \boldsymbol{z}$. (Kuvauksen s arvon $s(\boldsymbol{x}, \boldsymbol{x})$ muoto on hankalampi.)

Lause 5.13 (Metropolis–Hastings-ketjun tasapainojakauma)**.**
Metropolis–Hastings-algoritmilla 5.11 tuotetun Markovin ketjun tasapainojakauma on π.

Todistus. Näytetään että algoritmilla muodostettu Markovin ketju toteuttaa käänteyvyysehdon 5.8.

Jos $\boldsymbol{x} = \boldsymbol{z}$, niin selvästi pätee $\pi(\boldsymbol{x}) \cdot s(\boldsymbol{x}, \boldsymbol{z}) = \pi(\boldsymbol{z}) \cdot s(\boldsymbol{z}, \boldsymbol{x})$.

Oletetaan sitten $\boldsymbol{x} \neq \boldsymbol{z}$:

$$\pi(\boldsymbol{x}) \cdot s(\boldsymbol{x}, \boldsymbol{z})$$

$$
\begin{aligned}
&= \pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x}) \cdot \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z})}{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x})}\right\} \\
&= \min\left\{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x}),\ \pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z})\right\} \\
&= \min\left\{\pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z}),\ \pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x})\right\} \\
&= \pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z}) \cdot \min\left\{1,\ \frac{\pi(\boldsymbol{x}) \cdot q(\boldsymbol{z} \mid \boldsymbol{x})}{\pi(\boldsymbol{z}) \cdot q(\boldsymbol{x} \mid \boldsymbol{z})}\right\} \\
&= \pi(\boldsymbol{z}) \cdot s(\boldsymbol{z}, \boldsymbol{x})
\end{aligned}
$$

Lauseen 5.9 perusteella π on ketjun tasapainojakauma. □

Jos Markovin ketju on yhtenäinen, niin kääntyvyysehto varmistaa, että Markovin ketju konvergoituu kohti yksikäsitteistä tasapainojakaumaa π. On siis varmistettava, että ehdotusjakaumien q valinta ei muuta ketjua epäyhtenäiseksi. Yhtenäisyys on taattu, jos siirtymäkuvaus s mahdollistaa (usean askeleen) siirtymät kaikkien tilaparien välillä positiivisella todennäköisyydellä.

Metropolis–Hastings-algoritmilla tuotetun Markovin ketjun jokaista tilaa voidaan pitää satunnaisvektorina annetusta todennäköisyysjakaumasta π, kunhan konvergoituminen on varmistettu (katso luku 5.5.2). Ketjun peräkkäiset tilat eivät ole riippumattomia toisistaan, mutta tämä oli sallittua Monte Carlo -integroinnissa. Riippuvuutta kannattaa silti joskus vähentää, kuten myöhemmin tehdään ns. ohentamisen yhteydessä.

Oletetaan että Markovin ketjun nykyinen tila on ilmaistavissa reaaliluvulla x. Tilan ehdotusjakauma voidaan valita niin, että se on symmetrinen ja sen odotusarvo on nykyinen tila x, esimerkiksi Normaali$(x, 1^2)$. Ehdotusjakaumasta poimittu ehdotus on muotoa $x + \Delta x$, jossa *siirtymä* Δx ei ole itseisarvoltaan kovin suuri. Jos Metropolis–Hastings-algoritmissa käytetään tällaista ehdotusjakaumaa, niin Markovin ketjun käyttäytymistä kutsutaan *satunnaiskuluksi* (engl. random walk).

Metropolis–Hastings-algoritmi on itse asiassa yleistys varhaisemmasta Metropolis-algoritmista [MRR$^+$53]. Alkuperäisessä Metropolis-algoritmissa vaadittiin aiemmin mainittujen ehtojen lisäksi, että käytetyt ehdotusjakaumat q ovat symmetrisiä eli $q(\boldsymbol{z} \mid \boldsymbol{x}) = q(\boldsymbol{x} \mid \boldsymbol{z})$ kaikilla $\boldsymbol{x}, \boldsymbol{z} \in \Omega$. Tästä seuraa, että algoritmin 5.11 hyväksymistodennäköisyys (5.12) sievenee Metropolis-algoritmin tapauksessa muotoon

$$
\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{z})}{\pi(\boldsymbol{x})}\right\}.
$$

Jos ehdotetun tilan $\boldsymbol{z}$ todennäköisyystiheys $\pi(\boldsymbol{z})$ on ainakin yhtä suuri kuin nykyisen tilan eli $\pi(\boldsymbol{z}) \geqslant \pi(\boldsymbol{x})$, niin Metropolis-algoritmi siirtyy uuteen ehdotettuun tilaan varmasti.

Esitetystä algoritmista 5.11 on syytä huomioida, että se ei käytä todennäköisyystiheyksiä $\pi(\boldsymbol{x})$ suoraan, vaan niiden suhteita. Tästä syystä algoritmi voi käyttää tarkkojen todennäköisyystiheyksien sijaan mitä tahansa toisiinsa verrannollisia arvoja, esimerkiksi bayesiläisten mallien todennäköisyyksiä. Juuri tämä ominaisuus mahdollistaa (riippuvien) satunnaisvektorien poimimisen lähes mielivaltaisten ulottuvuuksien todennäköisyysjakaumista.

5.4.4 Gibbsin poiminta-algoritmi

Seuraavaksi esitellään *Gibbsin poiminta-algoritmi*[3] [Nea93, luku 4.1][GG84], jota voidaan pitää Metropolis–Hastings-algoritmin erikoistapauksena. Ehdotettava tila poimitaan mielivaltaisen ehdotusjakauman sijaan tasapainojakauman π ehdollisesta jakaumasta. Nämä ehdolliset jakaumat määrittävät tasapainojakauman π yksikäsitteisesti.

Palautetaan mieleen, että Markovin ketjun tila voidaan kuvata m-ulotteisena vektorina $\boldsymbol{x} \in \mathbb{R}^m$ jollakin m. Valitaan Markovin ketjun tila-avaruudelle erillisten *komponenttien* lukumäärä k, jossa $1 \leqslant k \leqslant m$. Olkoon $K = \{1, \ldots, k\}$ näiden komponenttien indeksien joukko. Erotellaan tilasta $\boldsymbol{x}$ yhteensä k erillistä komponenttia $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_k})$, jossa kullakin komponentilla i on yksi- tai moniulotteinen arvovektori $\boldsymbol{x_i}$. Merkitään vielä $\boldsymbol{x_{-i}} = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_{i-1}}, \boldsymbol{x_{i+1}}, \ldots, \boldsymbol{x_k})$ vektoria, josta puuttuu komponentin i arvovektori $\boldsymbol{x_i}$. Gibbsin poiminnan yksi askel on kuvattu algoritmissa 5.14. Jos (sallittu) alkutila $X^{(0)}$ on annettu, Markovin ketjun tilat $X^{(1)}, X^{(2)}, \ldots, X^{(n)}$ saadaan toistamalla Gibbsin algoritmia n kertaa.

Algoritmi 5.14 alustaa tilaan $\boldsymbol{z}$ ensin samat arvot, mitkä ovat Markovin ketjun nykyisellä tilalla $\boldsymbol{x}$. Algoritmi tuottaa tilan $\boldsymbol{z}$ jokaisen komponentin i arvovektorille $\boldsymbol{z_i}$ uudet arvot muiden komponenttien arvoilla ehdollistetusta todennäköisyysjakaumasta $\pi(\cdot \mid \boldsymbol{x_{-i}})$. Komponentin uudet arvot tallennetaan heti tilaan $\boldsymbol{z}$, joten uudet arvot otetaan huomioon seuraavan komponentin ehdollistetussa jakaumassa. Kun

[3] Gibbsin poiminta-algoritmin toiminta ei liity suoraan Gibbsin satunnaiskenttiin. Gibbsin poimintaa kutsutaan usein *Gibbsin otannaksi*.

Algoritmi 5.14 Gibbsin poiminta

Syöte: komponenttijoukko K, Markovin ketjun tila $X^{(t)} = \boldsymbol{x}$ sekä todennäköisyysjakauman π ehdolliset jakaumat $\pi(\cdot \mid \boldsymbol{z_{-i}})$ kaikilla $i \in K$

Tulos: Markovin ketjun tila $X^{(t+1)}$

1: $\boldsymbol{z} \leftarrow \boldsymbol{x}$
2: **toista jokaiselle** komponentille $i \in K$
3: $\quad \boldsymbol{y_i} \leftarrow \textsc{Satunnainen}(\pi(\cdot \mid \boldsymbol{z_{-i}}))$ ▷ Poimitaan $\boldsymbol{y_i}$ ehdollisesta jakaumasta
4: $\quad \boldsymbol{z} \leftarrow (\boldsymbol{z_1}, \ldots, \boldsymbol{z_{i-1}}, \boldsymbol{y_i}, \boldsymbol{z_{i+1}}, \ldots, \boldsymbol{z_k})$ ▷ päivitetään komponentti i
5: **lopeta toisto**
6: **palauta** $\boldsymbol{z}$ ▷ hyväksytään aina uudeksi tilaksi

kaikkiin tilan $\boldsymbol{z}$ komponentteihin on poimittu arvot, niin $\boldsymbol{z}$ on Markovin ketjun seuraava tila.

Lause 5.15 (Gibbsin tasapainojakauma)**.**
Gibbsin algoritmilla 5.14 tuotetun Markovin ketjun tasapainojakauma on π.

Todistus. Osoitetaan, että Gibbsin algoritmi on Metropolis–Hastings-algoritmin erikoistapaus.

Komponentin i arvovektorin päivityksessä käytettävä ehdotusjakauma q_i on seuraava:

$$q_i(\boldsymbol{z} \mid \boldsymbol{x}) = \begin{cases} \pi(\boldsymbol{z_i} \mid \boldsymbol{x_{-i}}) & \text{, jos } \boldsymbol{x_{-i}} = \boldsymbol{z_{-i}} \\ 0 & \text{muuten.} \end{cases} \tag{5.16}$$

Olkoot $\boldsymbol{x}$ ja $\boldsymbol{z}$ sellaiset tilat, että ne poikkeavat toisistaan vain komponentin i arvojen kohdalla eli $\boldsymbol{x_{-i}} = \boldsymbol{z_{-i}}$. Metropolis–Hastings-algoritmin hyväksymistodennäköisyys (5.12) komponentin i muutokselle eli siirtymälle tilasta $\boldsymbol{x}$ tilaan $\boldsymbol{z}$ on

$$\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot q_i(\boldsymbol{x} \mid \boldsymbol{z})}{\pi(\boldsymbol{x}) \cdot q_i(\boldsymbol{z} \mid \boldsymbol{x})}\right\}.$$

Ehdotusjakauma q_i on Gibbsin algoritmissa ehdollistettu jakauma π, joten saadaan

$$\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot \pi(\boldsymbol{x_i} \mid \boldsymbol{z_{-i}})}{\pi(\boldsymbol{x}) \cdot \pi(\boldsymbol{z_i} \mid \boldsymbol{x_{-i}})}\right\}.$$

Tiedetään että pätee $\boldsymbol{x_{-i}} = \boldsymbol{z_{-i}}$. Käytetään lisäksi ehdollisen todennäköisyyden määritelmää, jotta saadaan

$$\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{z}) \cdot \pi(\boldsymbol{x})/\pi(\boldsymbol{z_{-i}})}{\pi(\boldsymbol{x}) \cdot \pi(\boldsymbol{z})/\pi(\boldsymbol{x_{-i}})}\right\}.$$

Sievennetään ensin toisensa kumoavat termit pois. Kun käytetään jälleen tietoa $\boldsymbol{x_{-i}} = \boldsymbol{z_{-i}}$, niin saadaan tulokseksi

$$\alpha(\boldsymbol{z} \mid \boldsymbol{x}) = \min\left\{1,\ \frac{\pi(\boldsymbol{x_{-i}})}{\pi(\boldsymbol{z_{-i}})}\right\} = \min\{1, 1\} = 1.$$

Komponentin i ehdotusjakaumasta q_i poimitut ehdotukset hyväksytään siis varmasti. Gibbsin algoritmissa voidaan siis käydä kerralla kaikki komponentit läpi, koska jokaisen yksittäisen komponentin ehdotus hyväksytään. Gibbsin algoritmi voidaankin nähdä Metropolis–Hastings-algoritmin erikoistapauksena, jossa lopullinen ehdotus muodostetaan komponentti kerrallaan ja hyväksytään varmasti. Gibbsin algoritmin ehdotusjakauma q on siis yhdistelmä komponenttien ehdotusjakaumista q_i. Koska Metropolis–Hastings-algoritmilla tuotetulla Markovin ketjulla on tasapainojakauma π, niin tämä pätee myös Gibbsin algoritmille. □

Jotta Gibbsin algoritmilla tuotettu Markovin ketju konvergoituisi kohti tasapainojakaumaa, ketjun täytyy olla yhtenäinen ja jaksoton. Ehdollisesta jakaumasta poimittu ehdotus voi olla sama kuin nykyinen tila, joten jaksottomuus pätee aina. Yhtenäisyys vaatii, että ketjun siirtymäkuvaus sallii siirtymän mistä tahansa tilasta minkä tahansa tilajoukon johonkin tilaan. Tämäkin ehto toteutuu, sillä Gibbsin algoritmi käy läpi kaikkien komponenttien ehdolliset jakaumat: kaikki mahdolliset tilat ovat siis saavutettavissa.

Gibbsin algoritmi voidaan aina vaihtaa yleisempään Metropolis–Hastings-algoritmiin [Nea93, luku 4]. Tämä tarkoittaa, että algoritmin 5.14 minkä tahansa komponentin i päivitys voidaan toteuttaa Metropolis–Hastings-algoritmilla. Silloin komponentin i ehdotus poimitaan jostakin vain komponenttia i muuttavasta ehdotusjakaumasta q_i ja ehdotus hyväksytään hyväksymistodennäköisyydellä (5.16). Algoritmeja on siis mahdollista käyttää yhdessä. Tässä on varmistettava, että ehdotusjakaumat q_i pitävät Markovin ketjun yhtenäisenä.

Algoritmissa 5.14 voidaan päivittää kaikkia komponentteja kiinteässä järjestyksessä, mutta arvottukin järjestys toimii. Päivitettävä komponentti voidaan myös arpoa esimerkiksi k kertaa kaikkien komponenttien joukosta, jolloin kaikkia komponentteja ei välttämättä päivitetä kierroksen aikana lainkaan. On myös sallittua päivittää joitakin komponentteja useammin kuin muita. Valinnalla voi olla merkitystä, sillä komponenttien päivitysjärjestys ja -tiheys vaikuttavat ketjun sekoittumiseen (lisää luvussa 5.5.3).

Gibbsin algoritmi on Metropolis–Hastings-algoritmia tehokkaampi siinä mielessä, että jokainen ehdotus johtaa hyväksyntään. Ongelmana on, että komponenttien ehdollistetut jakaumat voivat olla hyvin hankalia. Jos jollakin komponentilla on erityisen hankala ehdollinen jakauma, niin komponentin päivittämisessä kannattaakin käyttää Metropolis–Hastings-algoritmia. Gibbsin algoritmilla on sekin etu, että sen ehdotusjakaumat määräytyvät suoraan tasapainojakaumasta π, kun taas Metropolis–Hastings-algoritmissa ohjelmoijan täytyy valita ne itse.

5.4.5 Muita poiminta-algoritmeja

Metropolis–Hastings- ja Gibbsin algoritmit ovat kaikkein suosituimpia menetelmiä satunnaisotosten poimimiseksi Markovin ketjun avulla. Algoritmeja on kuitenkin olemassa monia muitakin, ja seuraavaksi esitellään niistä lyhyesti kolme.

Viipalepoiminta-algoritmi (engl. slice sampling) [Nea03] on varteenotettava vaihtoehto silloin, kun Gibbsin algoritmin soveltaminen on hankalaa. Oletetaan että tarkoituksena on päivittää tilavektorista $\boldsymbol{x}$ vain yhtä komponenttia i. Olkoon kuvaaja f verrannollinen tasapainojakaumaan eli $f \propto \pi$. Arvotaan ensin reaaliluku y tasaisesti avoimelta väliltä $]0, f(\boldsymbol{x_i} \mid \boldsymbol{x_{-i}})[$. Sen jälkeen valitaan jollakin käytännöllisellä menetelmällä komponentin i uusi arvovektori $\boldsymbol{x_i}$ joukosta $\{\boldsymbol{x_i} \mid f(\boldsymbol{x_i} \mid \boldsymbol{x_{-i}}) > y\}$. Yksiulotteisessa tapauksessa y-arvon voi tulkita pystyakselin arvoksi ja x_i-arvon vaakaakselin arvoksi, joiden määrittämä piste (x_i, y) pysyy kuvaajan f (ja verrannollisesti tasapainojakauman π) alla.

Markovin ketjujen *pariutuksessa* (engl. coupling) [PW96] muodostetaan kaksi Markovin ketjua, joiden alkutilat ovat mielivaltaiset. Ideana on, että ajan mittaan ketjujen tilat alkavat muistuttaa toisiaan yhä enemmän, riippumatta alkutiloista. Kun ketjut lopulta *pariutuvat* (ovat samassa tilassa samalla hetkellä), niin ketjujen tulkitaan konvergoituneen (konvergoitumisesta puhutaan lisää luvussa 5.5.2). Markovin ketjujen pariuttaminen perustuu siihen, että ketjujen toiminta ei ole toisistaan riippumatonta: molemmat ketjut käyttävät samalla hetkellä samoja satunnaislukuja. Kun ketjut päätyvät lopulta samaan tilaan jollakin hetkellä, niin ketjujen toiminta on siitä eteenpäin identtistä. Jotta moniulotteiset tilat saataisiin pariutettua nopeasti samaan tilaan, niin ketjuja tulee ohjata toistensa kaltaisiksi, satunnaisuuden silti kärsimättä.

Pariutuspoiminta-algoritmi (engl. coupling from the past) [PW96] perustuu Marko-

vin ketjujen pariuttamiselle. Tavoitteena on poimia Markovin ketjusta satunnaisvektoreita, jotka ovat täsmälleen ketjun tasapainojakaumasta ilman riippuvuuksia. Tästä syystä algoritmia kutsutaankin *täsmällisen poiminnan algoritmiksi* (engl. exact, perfect sampling). Ideana on simuloida useita Markovin ketjuja jostakin menneisyyden hetkestä $-T$ alkaen ja tarkistaa, ovatko ketjut pariutuneet hetkeen 0 mennessä. Jos pariutuminen on tapahtunut, niin hetken 0 tila hyväksytään tasapainojakauman satunnaisvektoriksi; muuten ketjujen simulointi toistetaan aikaisemmasta hetkestä (esimerkiksi $-2T$) alkaen. Ketjujen alkutilat valitaan satunnaisesti ja menetelmä toistetaan jokaiselle poimittavalle satunnaisvektorille erikseen. Algoritmi tuottaa täsmälleen tasapainojakauman satunnaisvektoreita, mutta on laskennallisesti erittäin vaativa.

Joskus on vertailtavana useita malleja, joista halutaan selvittää ilmiötä parhaiten kuvaava malli. Eri mallit voivat esimerkiksi sisältää eri määrän muuttujia. Näissä tapauksissa on hyödyllistä käyttää reversible jump -algoritmia (RJMCMC) [Gre95][RC04, luku 11.2], joka vertailee mallin muuttujien arvojen lisäksi itse malleja keskenään. Tässä ei anneta yksityiskohtaista selvitystä algoritmin toiminnasta, vaan se kuvaillaan lyhyesti sanallisesti. Algoritmi on Metropolis–Hastings-algoritmin yleistys, jossa tilavektorin arvojen muuttumisen lisäksi myös tilavektorin pituus voi muuttua. Ulottuvuuksien lisäämiselle ja vähentämiselle lasketaan hyväksymistodennäköisyydet ja tarvittaessa nykyisen tila-avaruuden arvovektori kuvataan uudelle tila-avaruudelle. Algoritmin käyttäminen voi olla laskennallisesti erittäin vaativaa.

5.5 MCMC-menetelmien toteuttaminen

Jotta MCMC-menetelmän käyttäminen onnistuisi, toteuttamisvaiheessa on kiinnitettävä huomiota lukuisiin teknisiin yksityiskohtiin. MCMC-menetelmien toteuttaminen vaatii suurta huolellisuutta. Seuraavissa aliluvuissa käsitellään asioita, jotka vaikuttavat onnistumiseen.

5.5.1 Markovin ketjun sekoittuminen

Markovin ketjun sanotaan *sekoittuvan* (engl. mix) hyvin, jos sen perättäisten tilojen väliset riippuvuudet ovat pieniä ja tasapainojakaumaa voidaan approksimoida tarkasti jo vähäisellä määrällä ketjun peräkkäisiä tiloja. Markovin ketjun hyvä sekoittuminen on tärkeää, jotta ketjun tilojen voidaan katsoa poimitun halutusta tasa-

painojakaumasta. Huonosti sekoittuvista ketjuista poimitut satunnaisvektorit voivat nimittäin esiintyä eri tiheyksillä kuin tasapainojakaumassa. Parhaassa tapauksessa ketjun tilat ovat (lähes) riippumattomia satunnaisvektoreita ketjun tasapainojakaumasta.

Kun MCMC-menetelmän tuottamien toisistaan riippuvien satunnaisvektorien välistä riippuvuutta halutaan vähentää, niin käytetään *ohennettua* (engl. thinned) Markovin ketjua. Ohennetun ketjun tilat saadaan alkuperäisestä ketjusta jättämällä osa tiloista pois [Gam97, luku 5.3.1]. Ohennettu ketju voisi esimerkiksi sisältää joka kymmenennen tilan alkuperäisestä ketjusta – alkuperäisen ketjun muut tilat jätetään huomiotta. Ketjussa kaukana toisistaan olevat tilat eivät ole enää niin riippuvaisia toisistaan, joten niistä muodostettu ketju sekoittuu näennäisesti alkuperäistä paremmin. Ohennettu ketju on sekin Markovin ketju ja sillä on sama tasapainojakauma kuin alkuperäisellä ketjulla. *Ohennusparametriksi* kutsutaan lukua k, kun ohennettuun ketjuun valitaan alkuperäisestä ketjusta joka k:s tila. Menetelmä sopii kaikille Markovin ketjuille, mutta laskennallinen vaativuus kasvaa lineaarisesti ohennusparametrin k suhteen.

Markovin ketjun sekoittuminen riippuu lukuisista eri tekijöistä. Komponenttien ja ehdotusjakaumien valinnat ovat näistä tekijöistä tärkeimpiä ja ne käsitellään myöhemmin.

5.5.2 Markovin ketjun konvergoituminen

Otosten poiminta Markovin ketjun avulla lähtee aina liikkeelle alkutilan valitsemisesta. Alkutila voidaan valita kiinteäksi tai sitten se arvotaan jostakin todennäköisyysjakaumasta. Markovin ketjujen teoriassa on kyse siitä, konvergoituuko ketju kohti tasapainojakaumaa äärettömän mittaisessa ketjussa, riippumatta alkutilasta. Markovin ketjujen simulaatiot ovat kuitenkin rajallisia, joten herää kysymys: mistä tiedetään, onko ketju jo konvergoitunut?

Oletetaan että Markovin ketjulle valitulla alkutilalla on tasapainojakaumassa hyvin pieni todennäköisyystiheys. Jos ketju sekoittuu huonosti, niin voi kestää pitkään ennen kuin ketju siirtyy tiloihin, joilla on suuri todennäköisyystiheys. Jos Markovin ketjun simulointi keskeytetään liian aikaisin, niin ketjusta poimitut satunnaisvektorit eivät siten edusta tasapainojakaumaa. Erään Metropolis–Hastings-algoritmilla tuotetun Markovin ketjun kulku on kuvissa 5.17(a) ja 5.17(b).

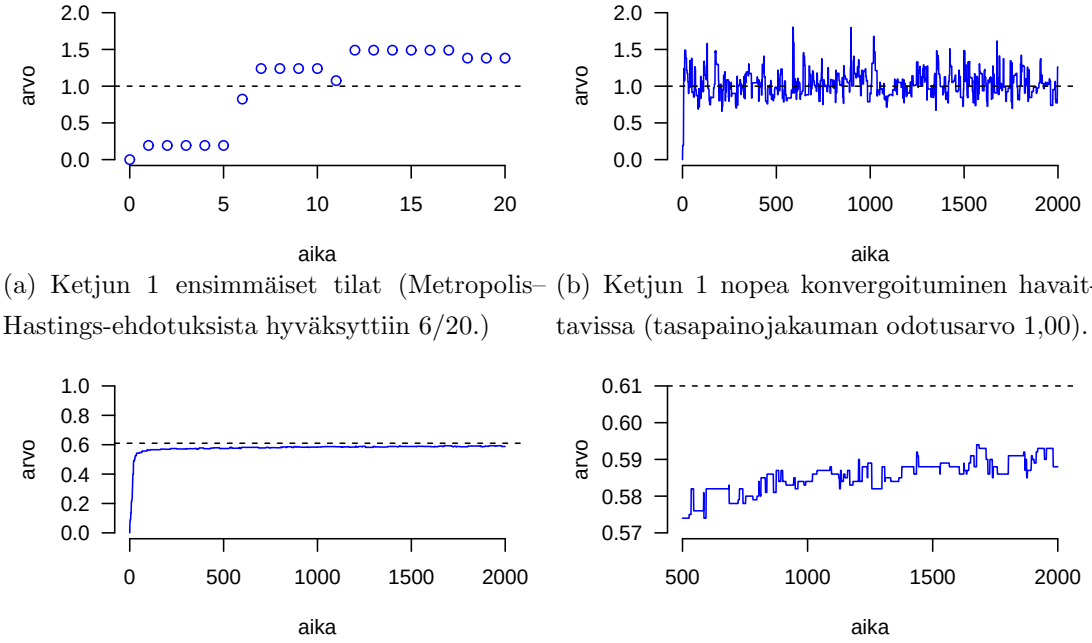
Markovin ketjun alkutilan vaikutus on poistettava ennen satunnaisvektorien poimimista. Tämä tapahtuu niin, että Markovin ketjun simulaatio-ohjelma käy ensin läpi *lämmittelyvaiheen* (engl. burn-in), jonka aikana se ei poimi satunnaisvektoreita. Lämmittelyvaihetta kestää, kunnes ketjun katsotaan konvergoituneen tasapainojakaumaan. Valitettavasti konvergoitumisen havaitseminen ei aina ole helppoa, kuten pian nähdään.

Lämmittelyvaihe lisää laskennallista rasitetta, sillä simulaatio-ohjelma ei kerää satunnaisvektoreita vaiheen aikana. Lämmittelyvaihe voidaan pitää lyhyenä, jos konvergoituminen tapahtuu nopeasti. Tähän vaikuttaa ketjun sekoittumisen lisäksi myös alkutilan valinta. Tilastotiede tarjoaa keinoja löytää estimaatteja, joiden avulla voidaan muodostaa sopiva alkutila. Kuvissa 5.17(a) ja 5.17(b) esiintyvän Markovin ketjun hyvä sekoittuminen hävittää alkuarvon merkityksen nopeasti.

Markovin ketjun konvergoitumisen varmistamiseen ei ole vuorenvarmaa menetelmää [CC96], sillä tasapainojakauman muoto on useimmiten tuntematon. Konvergoitumisen tutkimiseen on kehitetty lukuisia erilaisia menetelmiä, ja konvergoitumistesteissä on suotavaa käyttää mahdollisimman monia niistä. Analyysimenetelmät tarkkailevat joko simuloidun Markovin ketjun tiloja tai Markovin ketjun teoreettisia ominaisuuksia. Esimerkiksi R-kielellä toteutetut paketit CODA [BCV95] ja BOA [Smi05] sisältävät useita simuloidun ketjun analysointimenetelmiä, ja ovat osoittautuneet suosituiksi työkaluiksi Markovin ketjujen konvergoitumisen tutkimisessa. Seuraavaksi kuvataan lyhyesti muutama yksinkertainen analysointimenetelmä ja mainitaan vaihtoehtoisia menetelmiä.

Helpoin menetelmä ketjun konvergoitumisen tutkimiseksi perustuu kuvaajien visuaaliseen tarkasteluun [CC96]. Ketjun tiloja vastaavien arvovektoreiden yksittäisille arvoille piirretään jokaiselle omat kuvaajat, jossa x-akseli merkitsee ketjun hetkeä ja y-akseli arvoa tällä hetkellä. Jonkin hetken jälkeen kuvaajan käyttäytyminen ei enää muutu, vaikka ketjua jatketaan pidemmälle: tästä voi epäillä konvergoitumisen tapahtuneen. Katso esimerkit kuvista 5.17(a) ja 5.17(b). Tämä menetelmä kannattaa toistaa useilla erilaisilla Markovin ketjun alkutiloilla. Pelkkä kuvaajien tarkastelu saattaa johtaa harhaan: Markovin ketjun pidentäminen saattaa osoittaa, että kuvaajan muuttumattomalta vaikuttanut käyttäytyminen onkin vain hidasta muutosta kohti tasapainojakaumaa, kuten kuvista 5.17(c) ja 5.17(d) voi päätellä.

Gelmanin–Rubinin konvergoitumisen analyysimenetelmässä [BG98, GR92] simuloidaan useita Markovin ketjuja aloittaen eri alkutiloista. Ideana on laskea varianssit



(a) Ketjun 1 ensimmäiset tilat (Metropolis–Hastings-ehdotuksista hyväksyttiin 6/20.)

(b) Ketjun 1 nopea konvergoituminen havaittavissa (tasapainojakauman odotusarvo 1,00).

(c) Ketju 2 näyttää (virheellisesti) konvergoituneen.

(d) Ketju 2 vain sekoittuu hitaasti (tasapainojakauman odotusarvo 0,61)

Kuva 5.17: Esimerkkejä Markovin ketjuista ja niiden konvergoitumisesta.

Markovin ketjujen tiloille sekä yhden ketjun sisällä että useiden ketjujen välillä. Ensin valitaan jokin Markovin ketjun tasapainojakaumaa karkeasti approksimoiva jakauma, jonka varianssi on suurempi kuin tasapainojakaumalla. Tästä karkeasta jakaumasta poimitaan $m \geqslant 2$ kappaletta satunnaisvektoreita, jotka asetetaan Markovin ketjujen (m kappaletta) alkutiloiksi. On myös mahdollista valita toisistaan selvästi poikkeavat alkutilat käsin. Kaikkia Markovin ketjuja simuloidaan hetkeen $2n$ asti. Lämmittelyvaiheeksi määrätään hetket $0, \ldots, n$, joten vain tilat $n+1 \ldots 2n$ (eli n tilaa) otetaan analyysissä huomioon. Seuraavat vaiheet toistetaan ketjun tilojen jokaiselle ulottuvuudelle erikseen.

- Lasketaan jokaiselle simuloidulle Markovin ketjulle i sen arvojen $x_i^{(n+1)}, \ldots, x_i^{(2n)}$ aritmeettinen keskiarvo $\bar{x}_i$.
- Otetaan keskiarvo Markovin ketjujen otoskeskiarvoista: $\bar{x} = \frac{1}{m} \cdot \sum_{i=1}^{m} \bar{x}_i$.
- Markovin ketjujen otoskeskiarvojen välinen otosvarianssi $\frac{1}{n} \cdot B$ saadaan laskemalla $\frac{1}{n} \cdot B = \frac{1}{m-1} \cdot \sum_{i=1}^{m} (\bar{x}_i - \bar{x})^2$. Tämä kertoo ketjujen poikkeamat toisistaan.
- Lasketaan jokaiselle Markovin ketjulle i sen sisältämien arvojen otosvarianssi $s_i^2 = \frac{1}{n-1} \cdot \sum_{j=n+1}^{2n} (x_i^{(j)} - \bar{x}_i)^2$.

- Ketjujen saamien arvojen sisäisten varianssien keskiarvoksi tulee $W = \frac{1}{m} \cdot \sum_{i=1}^{m} s_i^2$. Tämä kertoo ketjujen sisäisen vaihtelun suuruuden.
- Markovin ketjun tasapainojakauman varianssin estimaatti on $\hat{V} = \frac{n-1}{n} \cdot W + \frac{1}{n} \cdot B$, joka on hieman todellista suurempi.
- Lopulta saadaan suhdeluku $\sqrt{\hat{R}} = \sqrt{\frac{\hat{V}}{W} \cdot \frac{df+3}{df+1}}$, jossa df on estimoitu vapausasteiden määrä.[4]

Suhdeluvusta $\sqrt{\hat{R}}$ voi päätellä, poikkeavatko eri ketjujen väliset arvot toisistaan yhtä paljon kuin yhden ketjun sisältämät arvot. Ketjut lähtevät liikkeelle eri alkutiloista, joten ketjujen väliset poikkeamat ovat aluksi suuria. Jos suhdeluku on lähellä ykköstä (esimerkiksi alle 1,10), niin kyseessä olevan muuttujan konvergoitumista hetkeen n mennessä ei voida kiistää [GCSR04, luku 11.6].

Jos tasapainojakauma on moniulotteinen, niin Gelmanin–Rubinin konvergoitumisanalyysi toistetaan jokaiselle muuttujalle. Ketjun pituutta $2n$ tulee kasvattaa niin suureksi, että jakauman kaikkien ulottuvuuksien kohdalla päädytään konvergoitumiseen. Menetelmä toimii parhaiten, kun tasapainojakauman yksittäisen ulottuvuuden reunajakauma on jotakuinkin normaalinen, mikä on hyvin yleistä. Suurin ongelma on, että moniulotteiselle jakaumalle tulisi simuloida lukuisia ketjuja, joka on laskennallisesti raskasta. Lisäksi alkutilojen valitseminen voi olla ongelma, sillä tasapainojakauman muoto on tavallisesti tuntematon ja käsin valitseminen mielivaltaista.

Geweken, Rafteryn–Lewisin ja Heidelbergerin–Welchin menetelmissä tutkitaan kaikissa vain yhden muuttujan konvergoitumista kerrallaan, kuten Gelmanin–Rubinin menetelmässäkin [CC96]. Erona on, että niissä tarkkaillaan useiden Markovin ketjujen sijaan vain yhtä ketjua. Tähän asti mainitut menetelmät sopivat kaikkiin MCMC-algoritmeihin, mutta jotkut muut menetelmät on kehitetty varta vasten Gibbsin algoritmilla tuotetun Markovin ketjun analysointiin. Joissakin menetelmissä konvergoituminen voidaan tarkistaa tasapainojakauman kaikille ulottuvuuksille kerralla, mutta ne ovat varsin monimutkaisia.

Aiemmin mainitut analysointimenetelmät käyttävät kaikki hyväkseen simuloidun Markovin ketjun tiloja. Perimmäisenä tarkoituksena on löytää lämmittelyjaksolle jokin päättymishetki, jonka jälkeen poimitut satunnaisvektorit ovat ketjun tasapainojakaumasta. Seuraavaksi esitettävän menetelmän avulla lämmittelyjaksolle on mahdollista löytää jokin pituus tutkimalla suoraan Markovin ketjun ominaisuuksia.

[4] Alkuperäisessä artikkelissa [GR92] esiintyvä kerroin $\frac{df}{df-2}$ on virheellinen [BG98].

Luvussa 5.4.5 esiteltiin Markovin ketjujen pariuttaminen. Kun kaksi eri tiloista aloittaneet ketjua pariutuvat, niiden voidaan ajatella molempien konvergoituneen. Markovin ketjujen pariuttamisen avulla voidaan saada arvioita lämmittelyvaiheen sopivalle pituudelle. Yksinkertaisten Markovin ketjujen tapauksessa odotusarvo pariutumiseen kuluvalle ajalle (mielivaltaisilla alkutiloilla) voidaan laskea analyyttisesti. Simuloidun ketjun tilojen sijaan käytetään ketjun siirtymätodennäköisyyksiä. Pariutumisanalyysi liioittelee lämmittelyjakson pituutta, ja analyysin toteuttaminen on hankalaa monimutkaisille Markovin ketjuille.

5.5.3 Komponentit

Gibbsin poiminta-algoritmissa 5.14 oletettiin, että Markovin ketjun tila-avaruuden ulottuvuudet oli jaettu komponentteihin. Yhdelle komponentille ehdotetaan uusia arvoja, kun muut komponentit pidetään ennallaan. Komponenttien valinta on hyvin tärkeä MCMC-simulaatiossa, sillä se vaikuttaa merkittävästi Markovin ketjun simuloinnin tehokkuuteen ja sekoittumiseen. Seuraava selvitys komponenttien valinnasta pätee myös Metropolis–Hastings-algoritmille.

Tarkastellaan tapausta, jossa on komponentti jokaista tila-avaruuden ulottuvuutta kohden. Jokainen komponentti sisältää silloin yksiulotteisen arvon. Tämän valinnan suurin etu on, että Gibbsin algoritmin tarvitsemat ehdollistetut jakaumat ovat yksinkertaisia yksiulotteisia jakaumia. Samaten Metropolis–Hastings-algoritmin komponenttikohtaiset yksiulotteiset ehdotusjakaumat on helpompi valita. Yhden komponentin päivittäminen onkin yleensä laskennallisesti tehokasta, sillä satunnaisvektorien tuottamiseksi yksiulotteisista jakaumista on tehokkaita algoritmeja.

Jos komponentit ovat vain yksiulotteisia, niin suuren mittakaavan muutokset voivat tapahtua hyvin hitaasti (huono sekoittuminen). Varsinkin jos joillakin ulottuvuuksilla on vahvoja riippuvuuksia keskenään, niin ulottuvuudet kannattaa sisällyttää samaan komponenttiin: niille ehdotetaan muutoksia yhdessä. Ketjun sekoittumisen kannalta moniulotteisen komponentin muutos on parempi kuin lukuisten yksiulotteisten komponenttien muutokset [Nea93, luku 4.4]. Jos komponentti on moniulotteinen, niin sen ehdotusjakauma tai ehdollistettu tasapainojakauma ovat myös moniulotteisia. Moniulotteista komponenttia kannattaa käyttää aina, kun päivityksessä vaadittavaa moniulotteista jakaumaa voidaan käsitellä tehokkaasti [Gam97, luku 5.3.5].

Oletetaan että MCMC-simulaattori yrittää päivittää jotakin moniulotteista komponenttia. Jos muilla komponenteilla ehdollistetusta (moniulotteisesta) jakaumasta voidaan poimia satunnaisotoksia tehokkaasti, niin komponentin päivittämisessä kannattaa käyttää Gibbsin algoritmia. Jos tämä on laskennallisesti liian työlästä, niin komponentin päivittämisessä voidaan käyttää Metropolis–Hastings-algoritmia. Jos Metropolis–Hastings-algoritmille ei löydy kunnollista ehdotusjakaumaa, niin komponentti on jaettava useampaan osaan. Kunnollisen ehdotusjakauman ominaisuuksia käsitellään seuraavassa aliluvussa.

5.5.4 Ehdotusjakaumat

Kun Markovin ketjun tila-avaruuden komponentit on valittu, Gibbsin algoritmiin 5.14 ei liity muita valintoja. Jos taas käytetään Metropolis–Hastings-algoritmia, niin jokaiselle päivitettävälle komponentille täytyy vielä valita ehdotusjakauma. Seuraavaksi käsitellään komponenttikohtaisia ehdotusjakaumia, mutta samat ajatukset pätevät myös alkuperäisessä algoritmissa 5.11. Ehdotusjakaumien valinta tulee tehdä niin, että Markovin ketju säilyy yhtenäisenä. Sen lisäksi ehdotusjakaumilla tulisi olla seuraavia ei-välttämättömiä, mutta toivottavia ominaisuuksia.

- (A) Komponentin i ehdotusjakauman q_i muodon tulisi vastata ehdollistettua tasapainojakaumaa $\pi(\cdot \mid \boldsymbol{x}_{-i})$ mahdollisimman hyvin. Katso kuvaa 5.3.
- (B) Ehdotusjakaumasta q_i pitäisi olla helppo poimia satunnaisvektoreita – laskennallisen tehokkuuden vuoksi.
- (C) Ehdotusten hyväksymistodennäköisyyksien tulisi olla riittävän suuria, jotta ketju sekoittuisi hyvin. Yksiulotteisilla komponenteilla esimerkiksi 40% ja moniulotteisilla komponenteilla 20% [GCSR04, 11.10].
- (D) Ehdotetun tilan tulisi poiketa nykyisestä riittävästi, jotta ketju sekoittuisi hyvin [Has70].

Ominaisuus (A) aiheuttaa ominaisuuksien (C) ja (D) täyttymisen. Ominaisuus (B) kuitenkin rajoittaa ehdotusjakaumien valintaa, joten ominaisuuden (A) toteuttaminen hankaloituu. Jos ominaisuutta (A) ei voida saavuttaa, niin ominaisuudet (C) ja (D) kilpailevat keskenään. Ominaisuutta (D) pidetään hieman tärkeämpänä kuin ominaisuutta (C) [Nea93, luku 4.4].

Jos ominaisuuden (A) toteuttavan jakauman keksiminen on erityisen vaikeaa, niin ehdotusjakauman voi valita satunnaiskuluksi [Nea93, luku 4.4]. Satunnaiskulun siirtymien suuruutta vaihtelemalla voi helposti säädellä hyväksymistodennäköisyyttä, sillä pienet siirtymät aiheuttavat suuria hyväksymistodennäköisyyksiä. Jos ominaisuus (D) ei toteudu, vaan siirtymät ovat liian pieniä, niin ketju sekoittuu huonosti. Ketjun sekoittuminen huononee eräässä mielessä neliöllisesti satunnaiskulun siirtymän lyhenemisen suhteen [Nea93, luku 4.4].

Ehdotusjakaumasta poimittu ehdotus ei aina kuulu muuttujan arvojoukkoon. Esimerkiksi normaalijakaumasta poimitut ehdotukset voivat olla negatiivisia, vaikka muuttujan arvojoukko sisältää vain positiiviset luvut. Arvojoukon ulkopuolisen ehdotuksen hyväksymistodennäköisyys on Metropolis–Hastings-algoritmissa nolla. Jos ehdotetut arvot ovat usein muuttujan arvojoukon ulkopuolella, niin ketju sekoittuu huonosti. Sekoittumista voi joskus parantaa niin, että muuttujan arvojoukon ulkopuoliset ehdotukset kuvataan arvojoukon ehdotuksiksi. Vanhan jakauman q pohjalta luodaan siis uusi ehdotusjakauma q'. Uuden jakauman q' todennäköisyystiheys arvolle z on silloin summa vanhan jakauman q niiden arvojen tiheyksistä, jotka kuvautuvat uuden jakauman q' ehdotukseksi z. Esimerkiksi negatiiviset luvut voisi kuvata positiivisiksi, jolloin todennäköisyystiheys, että nykyisestä tilasta x poimitaan (positiivinen) ehdotus z, on $q'(z \mid x) = q(z \mid x) + q(-z \mid x)$.

5.6 Bayesiläisten mallien käsittely

MCMC-menetelmän tutkimus lisääntyi selvästi, kun sen todettiin soveltuvan myös bayesiläisten mallien analyysiin [GS90]. Nykyisin bayesiläisten mallien analyysi on MCMC-menetelmien suosituin käyttökohde.

Bayesiläisten mallien yhteisjakaumat ovat laskettavissa tarkasti vain kaikkein yksinkertaisimmilla malleilla. Ennen kuin MCMC-menetelmä tuli yleisesti käyttöön, niin realististen bayesiläisten mallien analyysi oli erittäin vaikeaa [Nea93, luku 3.2]. Nykyisillä MCMC-menetelmillä voi käsitellä myös monimutkaisia posteriorijakaumia. On oikeutettua sanoa, että bayesiläinen mallintaminen on suosittua juuri MCMC-menetelmien vuoksi.

Hierarkkisilla bayesiläisillä malleilla oletettiin olevan ehdollinen riippumattomuus, kuten kaavassa (4.3). Olkoon $\mathcal{V}$ hierarkkisen mallin solmujen joukko. Verrataan arvoyhdistelmien yhteistodennäköisyystiheyksien suhdetta mallissa silloin, kun vain

muuttujan θ arvo poikkeaa. Havaitaan

$$\frac{\text{PR}(\theta = z,\ \mathcal{V} \setminus \{\theta\})}{\text{PR}(\theta = x,\ \mathcal{V} \setminus \{\theta\})} = (*)$$

$$(*) = \frac{\text{PR}(\theta = z \mid \texttt{vanhemmat}(\theta)) \cdot \prod_{V \in \mathcal{V} \setminus \{\theta\}} \text{PR}(V \mid \texttt{vanhemmat}(V) \setminus \{\theta\},\ \theta = z)}{\text{PR}(\theta = x \mid \texttt{vanhemmat}(\theta)) \cdot \prod_{V \in \mathcal{V} \setminus \{\theta\}} \text{PR}(V \mid \texttt{vanhemmat}(V) \setminus \{\theta\},\ \theta = x)}.$$

Supistetaan pois yhteiset tekijät. Muuttujan θ arvo vaikuttaa vain sen omaan ja sen lapsien prioritodennäköisyyksiin.

$$(*) = \frac{\text{PR}(\theta = z \mid \texttt{vanhemmat}(\theta)) \cdot \prod_{V \in \texttt{lapset}(\theta)} \text{PR}(V \mid \texttt{vanhemmat}(V) \setminus \{\theta\},\ \theta = z)}{\text{PR}(\theta = x \mid \texttt{vanhemmat}(\theta)) \cdot \prod_{V \in \texttt{lapset}(\theta)} \text{PR}(V \mid \texttt{vanhemmat}(V) \setminus \{\theta\},\ \theta = x)}. \tag{5.18}$$

Oletetaan että muuttujan θ päivitys tehdään Metropolis–Hastings-algoritmilla. Jos nykyinen arvo on x ja ehdotettu arvo z, niin algoritmi käyttää hyväkseen todennäköisyystiheyksien suhdetta (5.18). Kun lasketaan posteriorijakaumaa, niin normalisointivakiot supistuvat pois. Verrannolliset arvot riittävät posteriorijakauman tehokkaaseen approksimoimiseen MCMC-menetelmällä.

Jos muuttujan θ ehdollinen jakauma $\text{PR}(\theta \mid \mathcal{V} \setminus \{\theta\})$ on riittävän yksinkertainen, niin muuttujan θ päivitys voidaan toteuttaa tehokkaasti Gibbsin algoritmilla. Jakauma saadaan erityisen yksinkertaiseksi, jos muuttujien priorit valitaan sopivien liittojakaumien joukosta. Käytännössä Gibbsin algoritmia käytetään vain silloin, kun päivitettävällä muuttujalla on pieni diskreetti arvoalue tai sen priorina on liittojakauma.

Vaikka MCMC-menetelmät käsittelevät teoriassa kuinka monimutkaista jakaumaa tahansa, niin käytännössä simulaation vaatima aika tulee rajoitteeksi. Kuten yleensä mallintamisessa, mallin realistisuudesta on luovuttava sen verran, että mallin analysointi on laskennallisesti siedettävää.

MCMC-menetelmien ohjelmoiminen on virhealtista, sillä sen tuloksia ei voida aina varmentaa muilla menetelmillä – joskus MCMC-menetelmät ovat ainoa tapa saada ylipäätään mitään tuloksia. Virheiden välttämiseksi on järkevää pyrkiä automatisoimaan MCMC-menetelmän käyttö. Yksi merkittävä ohjelma on BUGS (Bayesian inference using Gibbs sampling) [STBG96], joka on suunniteltu suorittamaan MCMC-simulaatio bayesiläisille hierarkkisille malleille käyttäen Gibbsin algoritmia. BUGS-ohjelmalle annetaan syötteenä hierarkkisen mallin kuvaus ja mallin solmujen priorijakaumat. Koko MCMC-simulaattorin ohjelmoimisen vaiva vähentyy mallin solmujen ja niiden priorijakaumien ilmaisemiseen BUGS-ohjelman mallinkuvauskielellä. Erityisesti ohjelman GEOBUGS-versio soveltuu paikkatietoaineistojen käsitte-

lyyn. Aiemmin mainittu konvergoitumisen analysoimiseen tarkoitettu CODA-ohjelma kehitettiin alun perin juuri BUGS-ohjelman tulosteen analysoimiseen.

Toinen apuväline MCMC-simulaatioiden automatisointiin on BASSIST [TMSL98]. Se sallii myös Metropolis–Hastings-algoritmin käyttämisen, kuten BUGS-ohjelman uusimmat versiot.

Yleensä vain mallien muuttujien arvot ovat epävarmoja, mutta joskus myös mallin muuttujien määrään liittyy epävarmuutta. Bayesiläiset hierarkkiset mallit sallivat myös mallin ulottuvuuksien (muuttujien) määrän vaihtelun. Tämä sopii esimerkiksi silloin, kun selvitetään aikatietoaineistoista ne ajanhetket, joiden jälkeen aineiston arvot poikkeavat aiemmista. Tällainen ajanhetki viittaa yleensä tapahtumaan, jolla on merkittävä vaikutus mallinnettavaan ilmiöön. Siinä tapauksessa MCMC-simulaatiossa täytyy käyttää Metropolis–Hastings-algoritmin yleistystä: reversible jump -algoritmia.

Ennen kuin MCMC-menetelmät yleistyivät, mallin muuttujille oli tavallista laskea MAP-estimaatti. Tällä tavoin saadaan posteriorijakauman sijaan yksi arvoyhdistelmä. Jos posteriorijakauma on jo selvitetty, niin voidaan käyttää bayesiläisen mallin *MPM-estimaattia* (engl. maximum posterior mode) [Win95, luku 1.4], joka sekin antaa mallin muuttujien arvoyhdistelmän. Olkoon θ mallin jonkin muuttujan arvo ja $\boldsymbol{\theta}_{-}$ kaikkien muiden muuttujien arvovektori. MPM-estimaatissa mallin jokaisen muuttujan arvoksi θ tulee se arvo, joka maksimoi reunajakauman todennäköisyystiheyden

$$\operatorname*{argmax}_{\theta} \ \mathrm{PR}(\theta \mid \mathcal{D}) = \operatorname*{argmax}_{\theta} \int_{\boldsymbol{\theta}_{-}} \mathrm{PR}(\boldsymbol{\theta}_{-} \mid \mathcal{D}) \cdot \mathrm{PR}(\theta \mid \boldsymbol{\theta}_{-}, \mathcal{D}) \, d\boldsymbol{\theta}_{-}.$$

Muuttujien arvoiksi asetetaan siis niiden reunajakaumien todennäköisimmät arvot. MPM-estimaatin laskeminen vaatii posteriorijakauman reunatodennäköisyystiheyksien selvittämistä, joten sen laskeminen on työläämpää kuin MAP-estimaatin laskeminen.

5.7 Markovin satunnaiskenttien käsittely

Markovin satunnaiskenttä muodostaa yhteisjakauman, joten se on mahdollista liittää mukaan bayesiläiseen hierarkkiseen malliin. Markovin satunnaiskentän muuttujille asetetaan silloin vanhemmiksi kaikki sen naapurit satunnaiskentässä. Tässä tulee

ilmi Markovin ehdon laskennallinen hyödyllisyys: satunnaiskentän kaikkien muuttujien vaikutus saadaan aikaan ottamalla huomioon vain naapurit. Muuttujien arvojen ehdolliset jakaumat yksinkertaistuvat, joten Gibbsin algoritmin käyttäminen satunnaiskentän muuttujien arvojen päivittämisessä on tehokasta. Satunnaiskentän muuttujille voidaan tarvittaessa asettaa vanhemmiksi myös satunnaiskenttään kuulumattomia muuttujia, kuten tämän luvun lopussa näytetään.

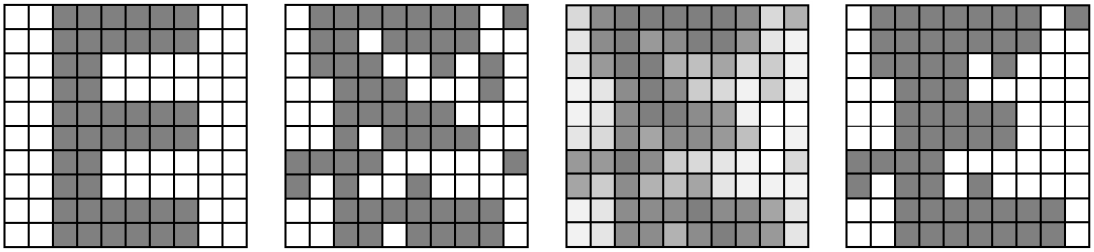
Todelliset aineistot ovat Markovin satunnaiskenttien tärkeimmät käyttökohteet. Niitä on silti hyödyllistä soveltaa myös keinotekoisille aineistoille, jos halutaan selvittää, miten uskottavia saadut tulokset ovat. Voidaan esimerkiksi tuottaa keinotekoinen aineisto ja poistaa siitä hieman dataa. Jälkikäteen voidaan verrata, miten hyvin Markovin satunnaiskentän avulla saadut tulokset ja alkuperäinen aineisto sopivat yhteen. Seuraavaksi käsitellään yksinkertainen kuvankäsittelytapaus – kattavaa vertailua ei ole mahdollista tehdä tässä tutkielmassa.

Esimerkki 5.19 (Ising-mallin sovellus)**.**
Seuraavaksi käytetään esimerkin 4.6 mallia korjaamaan vahingoittunutta mustavalkoista kuvaa. Vahingoittuneen kuvan pikseleiden arvot ovat vaihtuneet todennäköisyydellä 0,20*, joten pikseleiden tarkkuus on* $t_i = 0{,}80$ *kaikilla pikseleillä* i*. Ising-mallia on täsmennetty niin, että potentiaalifunktioissa ei suosita kumpaakaan arvoa eli* $\alpha_i = 0$ *kaikilla* $i \in I$*. Pikselien välisen vuorovaikutuksen käsin valittu voimakkuus on* $\beta_{i,j} = \beta = 0{,}80$ *kaikilla* $i, j \in I$*.*

Kuvassa 5.20 on esitetty kuvasarja, jossa vahingoittuneen kuvan (b) perusteella pyritään tuottamaan alkuperäinen kuva (a). Tässä on suoritettu MCMC-simulaatio yllä kuvatulle mallille. Kuvassa (c) on visualisointi pikseleitä vastaavien muuttujien posterioritodennäköisyyksille (eli reunajakaumien odotusarvoille). Kuvassa (d) on MPM-estimaatti eli kuvan (c) todennäköisyyksien luokittelu arvoiksi 0 *ja* 1*.*

Korjattu kuva muistuttaa alkuperäistä kuvaa enemmän kuin vahingoittunut kuva. Koska alkuperäinen kuva oli tuntematon, korjaus ei odotetusti pysty palauttamaan kirjainta E ehjäksi.

Käsitellään seuraavaksi tapausta, jossa Ising-mallin potentiaalifunktioille on annettu yhteinen vuorovaikutuskerroin eli $\beta_{i,j} = \beta$ kaikilla $i, j \in I$. Ajatus yleistyy kaikkiin Markovin satunnaiskenttiin, joiden potentiaalifunktiot sisältävät muuttujia. Muuttuja β ilmaisee lähialueiden välisen riippuvuuden voimakkuuden. Riippuvuuden voimakkuutta ei yleensä tunneta tarkasti, joten sitä on syytä käsitellä mallin muuttujana.



(a) Alkuperäinen kuva (b) Vahingoittunut kuva (kohinaa 20 %) (c) Posteriorijakauman reunajakaumien odotusarvot (d) Korjattu kuva, MPM-estimaatti

Kuva 5.20: Diskreetin kuvan korjaaminen Markovin satunnaiskentän avulla (MCMC ja Ising-malli). Valkoisen ruudun arvo on 0 ja tummanharmaan ruudun arvo 1.

Muuttujan β päivittäminen MCMC-simulaation aikana on laskennallisesti pulmallista. Oletetaan että nykyinen arvo on b ja ehdotusjakaumasta q_β poimittu arvo on b'. Silloin Metropolis–Hastings-algoritmin hyväksymistodennäköisyys on

$$\alpha(\beta = b' \mid \beta = b) = \min\left\{1,\ \frac{\text{PR}(\beta = b')}{\text{PR}(\beta = b)} \cdot \frac{\text{PR}(\boldsymbol{a} \mid \beta = b')}{\text{PR}(\boldsymbol{a} \mid \beta = b)} \cdot \frac{q_\beta(\beta = b \mid \beta = b')}{q_\beta(\beta = b' \mid \beta = b)}\right\},$$

jossa $\boldsymbol{a}$ on Markovin satunnaiskentän asetelma. Hyväksymistodennäköisyyden uskottavuudet $\text{PR}(\boldsymbol{a} \mid \beta = b')$ ja $\text{PR}(\boldsymbol{a} \mid \beta = b)$ eivät ole verrannollisia keskenään, sillä määritelmän 3.6 todennäköisyyskuvauksen normalisointivakion Z arvo riippuu muuttujan β arvosta. Muuttujan β tavallinen päivittäminen Metropolis–Hastings-algoritmilla vaatii, että normalisointivakio Z on laskettava sekä vanhalla että uudella muuttujan β arvolla.

Normalisointivakion voi laskea seuraavasti. Esitellään apumuuttuja, jonka ehdotusjakauma on valittu niin, että se poistaa normalisointivakion vaikutuksen [MGM06]. Ehdotusjakaumasta poimitaan satunnainen ehdotus pariutuspoiminta-algoritmilla, ja tämä ehdotus hyväksytään apumuuttujan uudeksi arvoksi Metropolis–Hastings-algoritmin hyväksymistodennäköisyyden mukaisesti. Menetelmän käyttäminen on teoreettisesti perusteltua, mutta laskennallisesti erittäin vaativaa.

Laskennallisista syistä tavallisen uskottavuuden sijaan käytetään usein *näennäisuskottavuutta* (engl. pseudo-likelihood) [Bes75], jossa Markovin satunnaiskentän arvot oletetaan riippumattomiksi toisistaan. Olkoot $\mathcal{X}$ Markovin satunnaiskenttään kuuluvien muuttujien joukko, S niiden indeksien joukko ja $\boldsymbol{a}$ Markovin satunnaiskentän asetelma. Silloin muuttujan β arvon b uskottavuustiheydeksi tulkitaan ehdollisista

todennäköisyyksistä muodostettu approksimaatio

$$\begin{aligned} \textsc{pr}(\boldsymbol{a} \mid \beta = b) &\approx \prod_{i \in S} \textsc{pr}(X_i = a_i \mid \beta = b,\ \mathcal{X} \setminus \{X_i\}) \\ &= \prod_{i \in S} \textsc{pr}(X_i = a_i \mid \beta = b,\ \mathcal{N}(X_i)). \end{aligned} \tag{5.21}$$

Tällä tavoin muodostettuja lukuja $\textsc{Pr}(\boldsymbol{a} \mid \beta = b')$ ja $\textsc{Pr}(\boldsymbol{a} \mid \beta = b)$ on mahdollista verrata toisiinsa. Ne tulkitaan toisiinsa verrannollisiksi uskottavuustiheyksiksi, jolloin muuttujan β päivittäminen Metropolis–Hastings-algoritmilla on mahdollista, vaikka ei erityisen tehokasta. Gibbsin algoritmin käyttäminen ei tule tässä kysymykseen, sillä muuttujan β ehdollisen jakauman selvittäminen on erittäin hankalaa. Näennäisuskottavuuden käyttäminen on käytännössä välttämätöntä, jotta pitkäjaksoinen MCMC-simulaatio olisi laskennallisesti realistinen. Näennäisuskottavuus ei ole valitettavasti teoreettisesti täysin perusteltu, mutta se tuottaa silti järkeviä tuloksia erityisesti Ising-mallin yhteydessä.

Tilastotieteessä on tavallisesti tyydytty laskemaan Markovin satunnaiskenttään kuuluville muuttujille MAP-estimaatti, koska se on laskennallisesti tehokas tuottaa. MAP-estimaatti eli kaikkein todennäköisin asetelma sisältää kiinteät arvot kaikille muuttujille, joten muuttujien arvoihin liittyvä epävarmuus häviää. Jos tuloksia on tarkoitus jatkokäsitellä, niin epävarmuuden sisältävä tieto kannattaa säilyttää. Jatkokäsittelyä varten on syytä selvittää satunnaiskentän kaikkien muuttujien reunajakaumat MCMC-menetelmän avulla.

6 Sovellukset ja testit

Markovin satunnaiskenttiä on käytetty aineistosta puuttuvan datan mallintamiseen monenlaisilla paikkatietoaineistoilla, ja pian esitellään joitakin näistä sovelluksista. Sen jälkeen sovelletaan bayesiläistä hierarkkista mallintamista ja MCMC-menetelmiä suomen kielen murresana-aineistolle ja Suomen lintujen pesimäaineistolle. Painotus on murresana-aineistossa. Lisäksi pohditaan saatujen tulosten merkitystä.

6.1 Aiemmat sovellukset

Markovin satunnaiskenttien sovellukset perustuvat lähes kaikki kuvankäsittelyyn kehitettyihin menetelmiin. Menetelmiä on käytetty useisiin paikkatietoaineistoihin, ku-

ten arkeologisiin kaivaustuloksiin [BYM91] sekä tautien [BYM91], lintujen [HH94] ja kasvien [WH97] levinneisyyksien selvittämiseen. Samoja menetelmiä on käytetty myös aikatietoaineistojen esiintymävoimakkuuksien päättelemiseen [AH97]. Lähtökohtana on, että aineistosta todetaan puuttuvan jonkin verran dataa, jota sitten mallinnetaan Markovin satunnaiskenttien avulla. Mainituissa esimerkeissä selvitetään todelliset esiintymät havaintojen perusteella bayesiläisen päättelyn avulla.

Joissakin kuvankäsittelysovelluksissa [GG84, Win95] on haluttu selvittää vain mallin todennäköisin arvojakauma eli MAP-estimaatti. Tämä on sopinut hyvin esimerkiksi maastokuvien lohkomiseen [KAV05] ja aivojen magneettikuvien parantamiseen [ZBS01]. Tilastotieteessä itsekorrelaation mallintamiseen käytetään muitakin menetelmiä kuin Markovin satunnaiskenttiä, mutta ne eivät ole saaneet kunnolla jalansijaa muualla. Erityisesti tautien levinneisyyden selvittämiseen on kehitetty runsaasti muitakin menetelmiä [BRT05].

6.2 Murresana-aineisto

Seuraavissa aliluvuissa käydään läpi tutkielmassa esiteltyjen menetelmien soveltaminen suomen murresana-aineistolle. Ensin esitellään itse aineisto ja sille kehitetty malli. Sitten kuvataan suoritetut testit ja arvioidaan tulosten merkitystä. Lopuksi pohditaan vaihtoehtoja mallin parantamiseksi ja nyt saatujen tulosten hyödyntämiseksi.

6.2.1 Aineiston kuvaus

Suomen murteiden sanakirjan työstäminen alkoi jo vuonna 1896, ja sen tarkoituksena oli kattaa suomen kaikkien murteiden sanasto [Tuo89]. Suomen murteita todettiin käytettävän paitsi Suomen nykyisten rajojen sisällä, myös Ruotsin Länsipohjassa, Venäjälle toisessa maailmansodassa luovutetuilla Karjalan alueilla, Jäämeren rannalla Ruijan alueella ja Suomenlahden pohjukassa Inkerissä. Epärealistinen suunnitelma, taloudelliset vaikeudet ja pelko työn viivästymisestä (ja murteiden sekoittumisesta) aiheuttivat sen, että sanojen keräämisessä päätettiin keskittyä noin kuuteenkymmeneen kuntaan[5]. Mukaan valittiin ne kunnat, joiden sanaston ajateltiin edustavan eri murrealueita hyvin. Keräämistä jouduttiin lopulta jatkamaan 1970-luvulle asti.

[5] *Kunnilla* tarkoitetaan tässä tekstissä niitä alueita, joita 1900-luvun alussa kutsuttiin *pitäjiksi*.

Valittuihin kuuteenkymmeneen kuntaan lähetettiin koulutetut kerääjät, joiden tehtävänä oli selvittää kuntien murresanasto. Muissa kunnissa jouduttiin turvautumaan vapaaehtoisten maallikoiden kirjeenvaihtoon. Lopputuloksena syntyi joukko sanalippuja, joissa lueteltiin ainakin osa sanaa käyttävistä kunnista. Erillisiä sanoja kertyi lopulta n. 400 000 ja kuntakohtaisia levikkitietoja miljoonia. Aineiston muuttaminen elektroniseen muotoon on käynnissä – nykyinen elektroninen aineisto kattaa vain pienen osan kerätystä datasta.

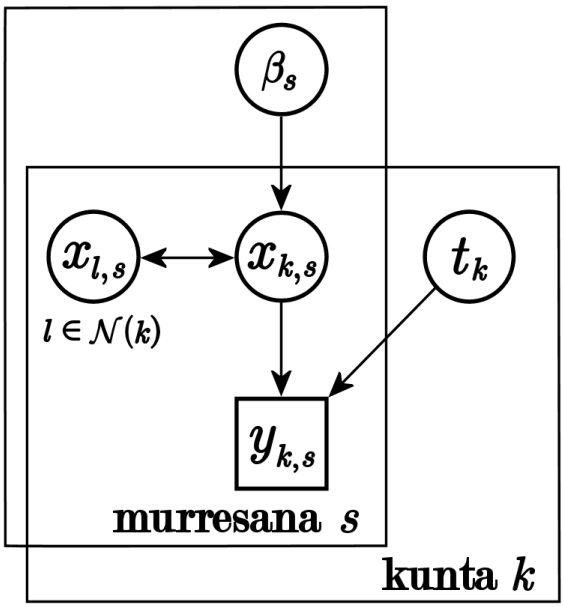
Tällä hetkellä käytettävissä olevassa elektronisessa aineistossa on 17100 murresanan levikkitiedot vuoden 1938 kuntajaolla, johon kuului 563 kuntaa. Aineistossa on jokaisen sana–kunta-parin kohdalla binäärinen arvo kertomassa sanan havaitsemisesta. Se, että sanaa ei ole havaittu, ei kuitenkaan tarkoita, ettei sitä käytettäisi. Kerättyjen sanojen määrän vaihtelu kunnittain on esitetty kuvassa 6.8(a) ja liitteen 3 kuvassa 2. Suomen kunnat ovat naapureita, jos niillä on yhteistä rajaa (käytettävässä aineistossa myös kulmittain kiinni olevat kunnat ovat naapureita). Testiin kuuluvien kuntien naapuriverkko on liitteen 3 kuvassa 1 (testiin kuulumattomat kunnat ovat ilman naapureita).

Aineiston perusteella eniten sanoja on kerätty juuri niistä kunnista, jotka asetettiin edustamaan murrealuettaan. Niistä kunnista, joiden sanasto on kerätty vapaaehtoisten voimin, puuttuu oletettavasti paljon havaintoja. Tätä puuttuvaa dataa mallinnetaan aiemmin esitellyillä menetelmillä niin, että lopputuloksena on joukko todennäköisyyksiä kuvaamassa sanojen käyttöä kunnissa. Tavoittena on luoda todennäköisyysaineisto, jonka tarkkuus on parempi kuin alkuperäisellä binäärisellä aineistolla.

6.2.2 Mallin kuvaus

Seuraavaksi esitellään bayesiläinen hierarkkinen malli murresanojen mallintamiseen, joka hyödyntää Markovin satunnaiskenttiä. Itse asiassa malli poikkeaa vain hieman esimerkissä 4.6 nähdystä, ja on hyvin samantyyppinen kuin eräs malli, jota on aiemmin käytetty lintujen levinneisyyden selvittämiseen [HH94]. Malli on esitetty kuvassa 6.1, ja sen sisältö käydään seuraavaksi läpi sanallisesti.

Binäärisen datamatriisin $\boldsymbol{Y}$ alkio $y_{k,s}$ kertoo, onko kunnassa k havaittu käytettävän sanaa s vai ei; binäärisen muuttujamatriisin $\boldsymbol{X}$ alkio $x_{k,s}$ kertoo, onko sana todellisuudessa käytössä kunnassa. Kuntakohtaisen muuttujavektorin $\boldsymbol{t}$ alkion t_k arvoväli



Kuva 6.1: Hierarkkinen bayesiläinen malli murresanoille.

on jatkuva $[0, 1]$. Muuttuja t_k kertoo, millä todennäköisyydellä jokin käytössä oleva sana havaittiin kunnassa k. Muuttujavektorin $\boldsymbol{\beta}$ reaaliarvoinen sanakohtainen alkio β_s säätää, kuinka voimakkaasti lähikunnat ovat samanlaisia sanan s osalta. Mallin kuvaus on seuraava:

$$\begin{aligned}
[\beta_s] &\sim \text{Tas}(0, 10), \\
[x_{k,s} \mid \beta_s,\ \mathcal{N}(x_{k,s})] &\sim \text{Bernoulli}(p_{k,s}), \text{ jossa } \text{logit}(p_{k,s}) = \beta_s \cdot \sum_{x_{j,s} \in \mathcal{N}(x_{k,s})} (2 \cdot x_{j,s} - 1), \\
[t_k] &\sim \text{Tas}(0, 1), \\
[y_{k,s} \mid x_{k,s},\ t_k] &\sim \text{Bernoulli}(x_{k,s} \cdot t_k).
\end{aligned} \tag{6.2}$$

Malliin sisältyy olettamus, että murresana-aineiston sanojen positiivisissa havainnoissa ei ole virheitä: havaittuja sanoja todella käytetään kunnissa. Joka tapauksessa virheitä olisi hankala tunnistaa ja niiden vaikutus voidaan olettaa merkityksettömäksi. Aiemmin datana käytetyt havaintojen tarkkuudet $\boldsymbol{t}$ ovat nyt kuntakohtaisia muuttujia, jotka arvioivat todennäköisyyksiä käytettyjen sanojen havaitsemiselle. Muuttujia siksi, että käytettävissä ei ole luotettavaa tietoa havaitsemisen tarkkuudesta eri kunnissa. Kuntakohtaisia siksi, että joitakin kuntia on tutkittu toisia enemmän. Kuntakohtaisista tarkkuusparametreista $\boldsymbol{t}$ ei ole tarkkaa etukäteistietoa, joten

priorijakaumaksi voidaan valita tasainen jakauma $\mathrm{Tas}(0,1)$. Lisäperusteena tasaisen jakauman valinnalle on, että sen käsittely on laskennallisesti tehokasta.

Kun todellinen käyttö $x_{k,s}$ ja havainnon tarkkuus t_k ovat tunnetut, niin havainto $y_{k,s}$ oletetaan ehdollisesti riippumattomaksi kaikista muista muuttujista. Tämä on luonteva oletus: kuntien ominaisuudet eivät vaikuta sanojen havaitsemiseen muissa kunnissa. Samalla oletetaan, että sanojen havaitsemisessa ei ole sanakohtaisia eroja – ainakaan yhtään ilmeistä syytä ei löydy. Todennäköisyydet voidaan ilmaista lausekkeella $\Pr(y_{k,s} \mid x_{k,s},\ t_k) = x_{k,s} \cdot t_k \cdot (2 \cdot y_{k,s} - 1) + (1 - y_{k,s})$, jossa muuttujat $y_{k,s}$ ja $x_{k,s}$ ovat binäärisiä ja t_k todennäköisyys. Seuraava taulukko on havainnollisempi.

$\Pr(y_{k,s} \mid x_{k,s},\ t_k)$		$x_{k,s}$ = 0	$x_{k,s}$ = 1
$y_{k,s}$	0	1,00	$1 - t_k$
	1	0,00	t_k

Tarkastellaan hetki vain yhtä sanaa s. Sisältäköön vektori $\boldsymbol{x_s} = (x_{1,s}, \ldots, x_{n,s})$ kaikki sanan s käyttöä kuvaavat muuttujat. Muuttujat $\boldsymbol{x_s}$ muodostavat oman Markovin satunnaiskentän, joka muistuttaa esimerkissä 4.6 käytettyä. Todennäköisyyskuvaus sanan s Markovin satunnaiskentälle on

$$\Pr(\boldsymbol{x_s} \mid \beta_s) = \frac{1}{Z} \cdot \exp\left(\beta_s \cdot \sum_{\{x_{i,s},\ x_{j,s}\} \in \mathcal{C}_2} 1_{x_{i,s} = x_{j,s}} \right),$$

ja sen ehdolliset todennäköisyydet saadaan mukaillen liitettä 1. Voidaan olettaa, että murresanat leviävät pitkälti naapurikuntien kautta sosiaalisen kanssakäymisen yhteydessä. Samaten esimerkiksi paikallisissa lehdissä ja tapahtumissa saatetaan käyttää paikallista kieltä. Lähikuntien välillä on siis riippuvuuksia, jotka voi ottaa huomioon mallissa. Tässä tapauksessa Ising-malli sopii hyvin riippuvuuksien mallintamiseen. Suurempien kuin kahden kokoisten klikkien tarkastelu vain monimutkaistaa Ising-mallia, eikä tee mallia juuri mielekkäämmäksi: vain kahden kokoiset klikit huomioidaan. Yhden kokoiset klikit jätetään huomiotta (asetetaan $\alpha_{k,s} = 0$), sillä aineisto ei sisällä murresanan esiintymän kanssa korreloivaa dataa samasta kunnasta. Aineiston jokaiselle sanalle asetetaan oma Markovin satunnaiskenttä, jonka kahden kokoisilla klikeillä on yksinkertaisuuden vuoksi yhteinen parametri β_s.

Tässä mallissa Markovin satunnaiskentän parametri β_s on vakion sijaan satunnaismuuttuja ja sanakohtainen. Satunnaisuus perustuu havaintoon, että ei ole yksinkertaista selvittää sopivaa itsekorrelaation vakiomäärää sanan esiintymien keskuudessa.

Sanakohtaisuus selittyy sillä, että yksi ainoa vakio β aiheuttaa erilaisen vaikutuksen eri sanojen mallintamisessa – on siis parempi määrittää β jokaiselle sanalle erikseen. Luvussa 6.2.4 kerrotaan lisää muuttujien β arvojen vaikutuksesta eri sanoihin. Sanakohtaisten muuttujien β_s suuruutta on hankala arvioida etukäteen, joten niiden priorijakaumaksi valitaan laskennallisesti tehokas tasainen jakauma Tas(0, 10). Niinkin suuri muuttujan β_s arvo kuin 10 on epätavallisen suuri ja aiheuttaa hyvin suuren samankaltaisuuden naapurikuntien välillä sanan s kohdalla.

Koska tarkkuusmuuttujien $\boldsymbol{t}$ arvot riippuvat mallissa kaikkien sanojen havainnoista, erilliset sanat eivät ole toisistaan riippumattomia. Kuvatulle mallille saadaan yhteisjakaumaksi

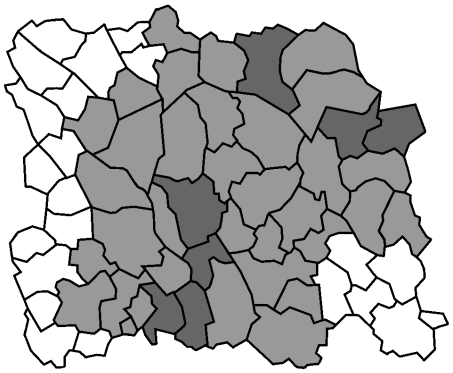
$$\begin{aligned} \text{PR}(\mathcal{V}) &= \text{PR}(\boldsymbol{t},\ \boldsymbol{\beta},\ \boldsymbol{X},\ \boldsymbol{Y}) = \text{PR}(\boldsymbol{t}) \cdot \text{PR}(\boldsymbol{\beta}) \cdot \text{Pr}(\boldsymbol{X} \mid \boldsymbol{\beta}) \cdot \text{Pr}(\boldsymbol{Y} \mid \boldsymbol{X},\ \boldsymbol{t}) \\ &= \left(\prod_k \text{PR}(t_k) \right) \cdot \prod_s \text{PR}(\beta_s) \cdot \text{Pr}(\boldsymbol{x_s} \mid \beta_s) \cdot \prod_k \text{Pr}(y_{k,s} \mid x_{k,s},\ t_k). \end{aligned} \tag{6.3}$$

Päämääränä on tuottaa todennäköisyyksiä suomalaisten murresanojen käytölle eri kunnissa. Ahvenanmaalla ja Vaasan seudulla vallitseva kieli on kuitenkin ruotsi, joten suomen kielen käyttämisestä ei juuri voida puhua. Nämä ruotsinkieliset kunnat eivät ole kiinnostavia suomen murteiden tutkimisessa: näihin kuntiin asetettiin todennäköisyydeksi nolla kaikkien sanojen käytölle ja niitä koskeva data poistettiin aineistosta. Kuntien naapurirakennetta muutettiin niin, että poistettavat kunnat eivät ole enää naapureita kiinnostavien kuntien kanssa. Tätä ratkaisua tukee sekin, että kyseisistä kunnista on vain vähän positiivisia sanahavaintoja. Vastaavista syistä aineistosta poistettiin lisäksi Inkerin ja Norjan alueiden kunnat.

Seuraavaksi esitellään menetelmä laskennan tehostamiseksi ja mallin parantamiseksi. Merkittävä osa aineiston murresanoista esiintyy vain muutamissa toisiaan lähellä olevissa kunnissa. Mitä kauempana jokin kunta on sanan havaituista esiintymistä, sitä epätodennäköisempää on sanan käyttö kunnassa. Esimerkiksi vain Karjalassa esiintyvä sana tuskin esiintyy Hämeessä, joten Hämeeseen kuuluvat kunnat kannattaa jättää simulaatiosta pois. Jos sanan s esiintyminen kunnassa k ei ole kovin uskottavaa, niin muuttujalle $x_{k,s}$ annetaan datana arvo nolla. Tällaista menettelyä kutsutaan tässä *paikalliseksi simulaatioksi*. Jäljempänä on kuvattu menettely sanalle s, joka toistetaan muillekin sanoille.

Paikallisessa simulaatiossa sanan s muuttujavektori $\boldsymbol{x_s}$ jaetaan kahteen osaan: *paikallisiin* muuttujiin $\boldsymbol{x_{s,+}}$ ja *kaukaisiin* $\boldsymbol{x_{s,-}}$. Olkoon $H = \{k \mid y_{k,s} = 1\}$ kaikkien

niiden kuntien joukko, joissa on tehty positiivinen havainto sanasta s. Kunnan k muuttuja $x_{k,s}$ kuuluu sanan s paikallisiin muuttujiin, jos ja vain jos kunnasta k on enintään ℓ askeleen mittainen polku (naapuriverkossa) johonkin joukon H kuntaan; muuten $x_{k,s}$ kuuluu sanan s kaukaisiin muuttujiin. Lukua ℓ kutsutaan paikallisen simulaation *kattavuudeksi*. Kuvassa 6.4 on esimerkki muuttujien jaosta paikallisiin ja kaukaisiin, kun käytetään paikallista simulaatiota kattavuudella kaksi.



Kuva 6.4: Paikallinen simulaatio (kattavuudella kaksi) jakaa kuntien muuttujat paikallisiin (harmaat) ja kaukaisiin (valkoiset) yhden sanan kohdalla. Positiiviset sanahavainnot on merkitty tumman harmaalla.

Sanan s kaukaisten muuttujien $\boldsymbol{x}_{\boldsymbol{s},-}$ arvoiksi annetaan paikallisessa simulaatiossa datana nollat. Paikallisten muuttujien $\boldsymbol{x}_{\boldsymbol{s},+}$ ja tarkkuusmuuttujien $\boldsymbol{t}$ päivitys hoituu silloinkin tavallisesti. Voimakkuusmuuttujan β_s päivityksessä käytettävä näennäisuskottavuus lasketaan paikallisessa simulaatiossa sekä paikallisten muuttujien $\boldsymbol{x}_{\boldsymbol{s},+}$ että kaukaisten muuttujien $\boldsymbol{x}_{\boldsymbol{s},-}$ arvoista. Kaukaiset muuttujat $\boldsymbol{x}_{\boldsymbol{s},-}$ ovat mallin dataa, ja muuttujan β_s päivityksen yhteydessä ne tulkitaan näennäisuskottavuuden laskuissa tavallisiksi muuttujiksi, joiden arvoina on aina nollia. Näennäisuskottavuus on joka tapauksessa pelkkä approksimaatio uskottavuudelle, joten juuri esitelty approksimaatiomenettely on sallittavissa.

6.2.3 Laskennalliset yksityiskohdat

Toteutettavana on simulaatio-ohjelma, joka selvittää murresanamallin yksittäisten sanojen käytön todennäköisyydet eri kunnissa. Ohjelma approksimoi aineiston datalla ehdollistettua posteriorijakaumaa poimimalla sopivasta Markovin ketjusta satunnaisvektoreita, ja käyttää Monte Carlo -menetelmää odotusarvojen laskemiseksi.

Sanojen käytön todennäköisyydet saadaan $\boldsymbol{X}$-muuttujien reunajakaumien odotusarvoista.

Tavallinen toteutusratkaisu olisi kuvata malli BUGS-ohjelman jonkin version mallinkuvauskielellä ja antamalla BUGS-ohjelman huolehtia satunnaisvektorien poimimisesta. Tämä ei kuitenkaan sovellu käytössä olevaan murresana-aineistoon, sillä aineiston koko on hyvin suuri. BUGS-ohjelman käyttäminen olisi joko mahdotonta tai liian hidasta. Murresanamallille nyt erikseen toteutettu ohjelma hyödyntää mallin laskennallisia ominaisuuksia ja on riittävän tehokas.

Kuvatulle murresanamallille toteutettiin C++ -kielellä erillinen MCMC-simulointiohjelma tätä tutkielmaa varten. Ohjelma käyttää sekä Gibbsin algoritmia että Metropolis–Hastings-algoritmia. Komponenteiksi valittiin yksittäiset muuttujat: matriisin $\boldsymbol{X}$ sekä vektorien $\boldsymbol{\beta}$ ja $\boldsymbol{t}$ kaikki alkiot. Yksi Markovin ketjun siirtymä sisältää jokaisen muuttujan arvon päivityksen aina samassa järjestyksessä. Jotta ohjelma pystyisi säilyttämään laskentatarkkuuden myös hyvin pienien todennäköisyyksien yhteydessä, se käyttää laskennassa todennäköisyyksien logaritmeja.

Muuttujamatriisin $\boldsymbol{X}$ jokainen alkio päivitetään erikseen Gibbsin algoritmilla. Kaikki tarvittavat ehdolliset todennäköisyydet saadaan ottamalla huomioon naapurimuuttujien arvot kuten ehdollisessa lausekkeessa (6.2).

Muuttujavektorin $\boldsymbol{t}$ jokainen tarkkuusmuuttuja päivitetään erikseen Metropolis–Hastings-algoritmilla. Tämän Metropolis-päivitysaskeleen ehdotusjakaumana on satunnaiskulku Normaali$(c,\ 0{,}07^2)$, jossa c on muuttujan nykyinen arvo. Ehdotusjakauman käyttäminen tuottaa hyväksymistodennäköisyydeksi noin 16 % (hajonta 4,3).

Markovin satunnaiskenttien voimakkuuksia säätelevien vektorin $\boldsymbol{\beta}$ alkioiden päivitys tapahtuu yksi kerrallaan Metropolis–Hastings-algoritmilla. Näiden muuttujien päivityksen yhteydessä joudutaan käyttämään tavallisen uskottavuuden asemesta näennäisuskottavuutta, kuten luvussa 5.7 todettiin. Metropolis–Hastings-algoritmin käyttämä ehdotusjakauma on satunnaiskulku Normaali$(c,\ 0{,}7^2)$, jossa c on muuttujan nykyinen arvo. Hyväksymistodennäköisyys on keskimäärin 31 % (hajonta 21).

MCMC-simulaattorin annettiin tuottaa Markovin ketjuun 110 000 tilaa, joista 10 000 ensimmäistä määrättiin lämmittelyvaiheeksi. Murremallin simulaatio kesti yhteensä 160 tuntia (laitteiston nopeus 3 GHz). Kun sovellettiin kattavuuden kolme paikallista simulaatiota, niin simulaation kesto oli 80 tuntia. Odotusarvojen laskemiseen

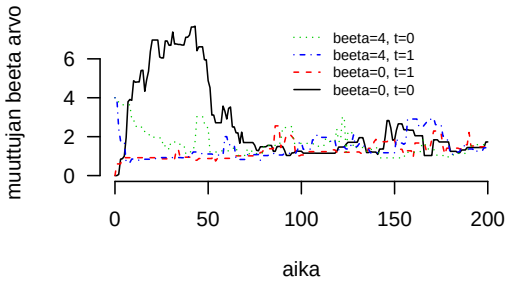
käytettiin ohennettua Markovin ketjua, jonka ohennusparametri oli 10. Muuttujien $\boldsymbol{t}$ ja $\boldsymbol{\beta}$ alkuarvoiksi valittiin 0,5 ja 1,0 (tällä ei ollut juuri merkitystä).

6.2.4 Tulokset

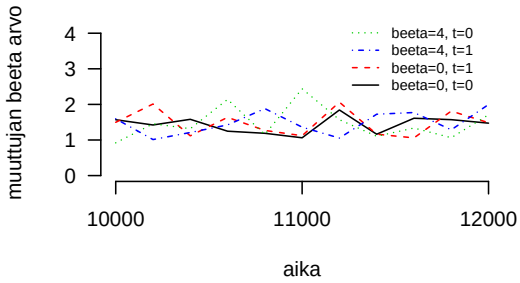
Markovin ketjun konvergoitumista kokeiltiin neljällä eri ketjulla, joista jokainen aloitti erilaisista alkutiloista. Ketjujen alkuarvot asetettiin valitsemalla, että muuttujilla $\boldsymbol{\beta}$ oli kaikilla sama alkuarvo 0,0 tai 4,0 ja muuttujilla $\boldsymbol{t}$ oli kaikilla sama alkuarvo 0,0 tai 1,0. Kuvassa 6.5 näkyy tyypillisten muuttujien β_{39} ja t_{313} (luultava) konvergoituminen. Molemmille muuttujille sovellettiin Gelmanin–Rubinin, Geweken, Heidelbergerin–Welchin ja Rafteryn–Lewisin konvergoitumisen analyysimenetelmiä – yksityiskohtaiset tulokset ovat liitteessä 2. Lyhyesti sanottuna muuttujan β_{39} konvergoitumista ei voi kiistää, ja lämmittelyvaihekin voidaan pitää lyhyenä. Muuttujan t_{313} konvergoituminen oli selvästi hitaampaa. Kun lämmittelyvaiheen kestoksi valittiin 8 000, niin analyysit eivät kiistäneet konvergoitumista. Simulaatiossa käytetty ohennusparametri 10 oli lähellä suositeltuja arvoja molempien muuttujien tapauksessa. Mainittujen kahden muuttujan tarkan konvergoitumisanalyysin lisäksi myös muutaman muun muuttujan konvergoituminen tarkastettiin visuaalisesti. Vaikuttaa siltä, että käytetty lämmittelyvaiheen pituus 10 000 on jopa hieman liioiteltu.

Muuttujia β_{39} ja t_{313} vastaaville Markovin ketjun arvoille laskettiin myös itsekorrelaation määrä sen mukaan, miten paljon ketjun arvot korreloivat saman ketjun muihin arvojen kanssa eri viiveillä. Tulokset ovat kuvassa 6.6 ja (yksityiskohtaisemmin) liitteessä 2. Simulaatiossa käytetty ohennusparametri 10 ei vielä poista itsekorrelaatiota täysin, mutta Monte Carlo -menetelmää voi silti käyttää. Itsekorrelaation määrää voi vähentää etsimällä parempi ehdotusjakauma.

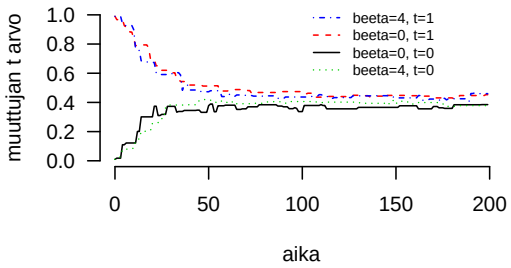
Muuttujien reunajakaumat ovat kiinnostavia, sillä niiden avulla on mahdollista kehittää parempia ehdotusjakaumia. Muuttujien β_{39} ja t_{313} reunajakaumien approksimaatiot ovat kuvassa 6.7. Muuttujan t_{313} reunajakauman approksimaatio muistuttaa normaalijakaumaa jopa niin paljon, että se läpäisi Shapiron–Wilkin normaalisuustestin. Muuttujan β_{39} arvot eivät sen sijaan läpäisseet testiä – testien yksityiskohdat ovat liitteessä 2. Voimakkuusmuuttujien $\boldsymbol{\beta}$ ehdotusjakaumiksi on silti aiemmin ehdotettu normaalijakaumaa [HH94]. Tarkkuusmuuttujat $\boldsymbol{t}$ ovat erittäin huipukkaita ja siistejä. Voimakkuusmuuttujien $\boldsymbol{\beta}$ reunajakaumat ovat sen sijaan laakeampia ja vaihtelevampia.



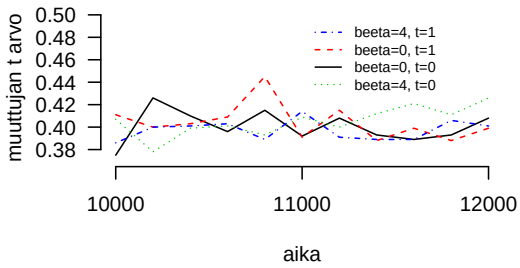
(a) Muuttujan β_{39} kuvaaja ketjun hetkillä 0–200.



(b) Muuttujan β_{39} kuvaaja ketjun hetkillä 10 000–12 000.

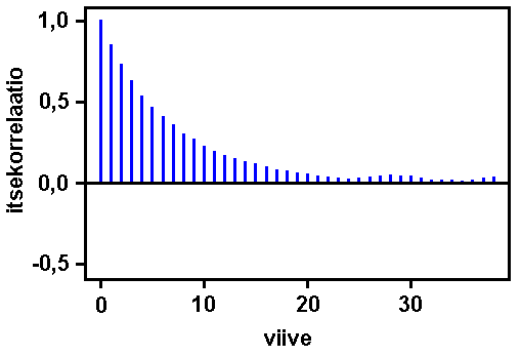


(c) Muuttujan t_{313} kuvaaja ketjun hetkillä 0–200.

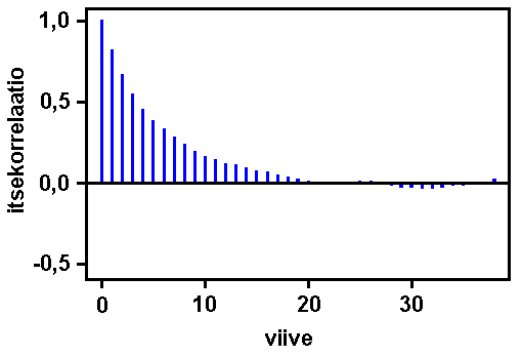


(d) Muuttujan t_{313} kuvaaja ketjun hetkillä 10 000–12 000.

Kuva 6.5: Kuvat muuttujien β_{39} ja t_{313} konvergoitumisesta. Kuvassa neljä eri Markovin ketjua, joissa muuttujien $\boldsymbol{\beta}$ alkuarvot olivat 0,0 tai 4,0 sekä muuttujien $\boldsymbol{t}$ alkuarvot 0,0 tai 1,0. Kuvien (b) ja (d) ketjuja on ohennettu parametrilla 200.

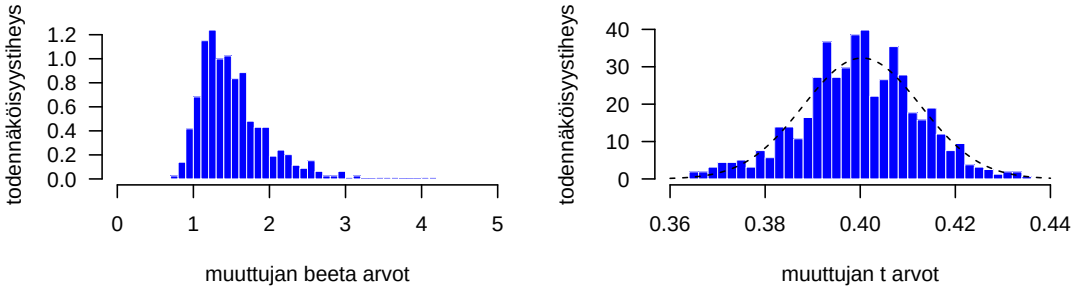


(a) Muuttujan β_{39} itsekorrelaatio.



(b) Muuttujan t_{313} itsekorrelaatio.

Kuva 6.6: Kuvat sekä voimakkuusmuuttujan β_{39} että tarkkuusmuuttujan t_{313} arvoissa havaitusta itsekorrelaatiosta. Itsekorrelaatio on laskettu ohentamattoman Markovin ketjun tiloista hetkillä 10 000–12 000.



(a) Muuttujan β_{39} reunajakauma. (Odotusarvo 1,50.)

(b) Muuttujan t_{313} reunajakauma ja sovitettu normaalijakauma. (Odotusarvo 0,40.)

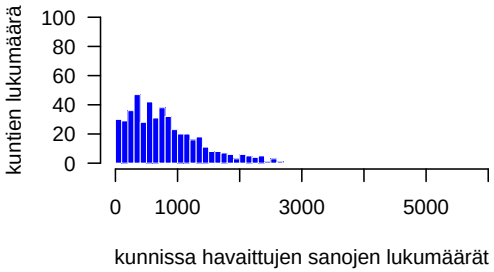
Kuva 6.7: Esimerkkikuvat tyypillisistä $\boldsymbol{\beta}$- ja $\boldsymbol{t}$-muuttujien reunajakaumista. Jakaumat muodostettiin Markovin ketjun tiloista hetkillä 10 000–17 900, ohennusparametrilla 10.

Liitteessä 3 on kuvat kuuden eri sanan havainnoista. Lisäksi liite sisältää kuvat $\boldsymbol{X}$-muuttujien reunajakaumien odotusarvoista eli sanojen käytön todennäköisyyksistä näiden sanojen kohdalla. Tulokset on laskettu sekä tavallisen simulaation että paikallisen simulaation (kattavuudella kolme) tuottamista posteriorijakaumista.

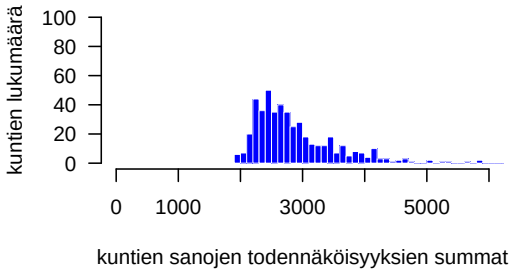
Tavallinen simulaatio tuottaa sanojen käytölle suurempia todennäköisyyksiä naapuriverkon reunoilla sijaiseville kunnille kuin verkon sisällä oleville, kuten nähdään liitteen 3 kuvasta 4 (tavallinen simulaatio). Tämän syyksi paljastuu, että naapurikuntien lukumäärä on reunalla vähäisempi, jolloin naapurikuntien kokonaisvaikutus on tavallista pienempi. Tätä ei-toivottavaa ilmiötä kutsutaan Ising-mallin *reunavääristymäksi*. Reunavääristymä toimii myös toiseen suuntaan: reunalla olevilla kunnilla voi olla turhan pieniä todennäköisyyksiä, vaikka naapureilla olisi suuria. Myöhemmin selvitetään, miksi reunavääristymän korjaaminen kuntien naapurien lukumäärän perusteella ei ole aivan yksinkertaista.

Utsjoelle (pohjoisessa), Keltolle (kaakossa) ja Merimaskulle (lounaassa) koituu liitteen 3 kuvassa 4 (tavallinen simulaatio) liioitellun suuret sanan käytön todennäköisyydet verrattuna sanan havaintoihin. Tämä johtuu osittain siitä, että β_{156}-muuttuja saa pieniä arvoja (odotusarvo 0,58) – vähäinen riippuvuus naapurikunnista suosii sanalle esiintymistodennäköisyyksiä, jotka ovat lähellä arvoa 0,5. Toinen syy on, että näillä kunnilla on kaikilla vain yksi naapurikunta, jolloin reunavääristymä voimistuu. Kolmanneksi, mainituilla kunnilla on lähinaapurustossa paljon sellaisia kuntia, joista on kerätty vain vähän sanoja – epävarmuus johtaa näissä tapauksissa todennäköisyyksien kasvamiseen.

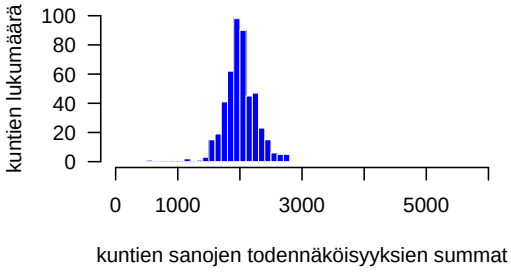
Kuvassa 6.8(a) on histogrammi kuntien alkuperäisten sanahavaintojen lukumääristä. Kuvissa 6.8(b) ja 6.8(c) on histogrammit sanojen käytön todennäköisyyksien summista kunnittain sekä tavallisella että paikallisella simulaatiolla (kattavuudella kolme). Jotkin tavallisen simulaation tuottamista todennäköisyyssummista ovat poikkeuksellisen suuria; paikallisesta simulaatiosta lasketut todennäköisyysmassat ovat lähellä parhaiten tutkittujen kuntien alkuperäisiä sanamääriä. Tavallisen simulaation suurimmat – ja luultavimmin epärealistisimmat – todennäköisyysmassat kuuluvat niille kunnille, joihin reunavääristymä vaikuttaa eniten.



(a) Murresanahavaintojen määrä.



(b) Todennäköisyysmassa tavallisella simulaatiolla.



(c) Todennäköisyysmassa paikallisella simulaatiolla kattavuudella kolme.

Kuva 6.8: Kunnittain havaittujen sanojen lukumäärä ja sanojen käytön todennäköisyyksien summa simulaatioiden jälkeen.

Paikallisen simulaation tuottamissa sanojen käytön todennäköisyyksissä ei ole havaittavissa kovin paljon reunavääristymää, sillä monien ongelmakuntien todennäköisyydet (nollat) on annettu datana. Paikallisen simulaation tuloksista liitteen 3 kuvissa on havaittavissa, että muuttujien $\boldsymbol{\beta}$ (odotus)arvot ovat pääsääntöisesti korkeampia kuin tavallisen simulaation tuottamat. Tämä johtuu siitä, että paikallisessa simulaatiossa sanan esiintymän arvo on määrätty datassa nollaksi hyvin monessa kunnassa, jolloin naapurikunnat muistuttavat toisiaan enemmän kuin tavallisessa simulaatiossa. Nollien välinen voimakas riippuvuus aiheuttaa mallissa myös voima-

kasta ykkösten riippuvuutta.

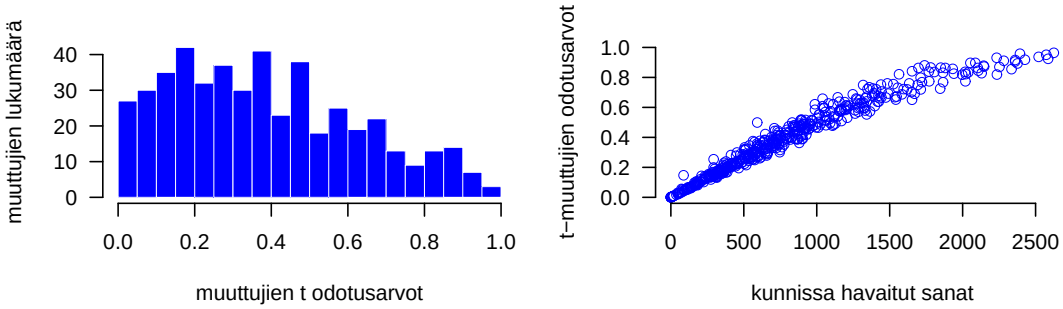
Paikallisen simulaation tulokset näyttävät uskottavammilta kuin tavallisen simulaation tulokset. Paikallisessa simulaatiossa on kuitenkin vielä yksi heikkous. Liitteen 3 kuvasta 2 nähdään, että naapuriverkossa aivan lounaassa olevan kunnan (Keltto) lähinaapuristossa on vain kuntia, joista on kerätty vähän sanoja. Kattavuuden kolme paikallinen simulaatio asettaa Keltolle useimpien sanojen käytölle todennäköisyydeksi nollan, sillä Keltosta ei ole kolmen mittaista polkua hyvin tutkittuihin kuntiin. Keltto erottuu pienenä palkkina todennäköisyyssummalla 596 kuvassa 6.8(c).

Pääsääntöisesti voidaan havaita, että sama muuttujan β arvo kasvattaa laajalle levinneiden sanojen käytön todennäköisyyksiä enemmän kuin niukasti levinneiden. Esimerkiksi liitteen 3 kuvissa 4 ja 8 paikallisten simulaatioiden todennäköisyydet muuttuvat eri tavalla, vaikka muuttujilla β_{156} ja β_{260} on lähes sama odotusarvo. Voimakkuusmuuttujien $\boldsymbol{\beta}$ arvot ovat suurimmillaan ryppäissä esiintyvillä vähän havaituilla sanoilla. Tälle vaihtelulle ei löytynyt helppoa selitystä, vaan se vaatii lisätutkimusta. Vaihtelusta johtuen malliin ei asetettu vain yhtä muuttujaa β, vaan malli sisältää naapuririippuvuutta säätelevän muuttujan jokaiselle sanalle.

Reunavääristymän vähentämiseksi on aiemmin kokeiltu seuraavaa yksinkertaista ratkaisua [Hög95, s. 75]. Markovin satunnaiskentän ehdollinen todennäköisyys kunnan k arvoille lasketaan kunnan k naapurien lukumäärällä jaetuista potentiaalifunktioiden arvoista. Testit murresanamallilla osoittivat, että menetelmä todella vähentää reunavääristymää. Syynä on, että naapurien kokonaisvaikutus pysyy kunnille samana riippumatta naapurien lukumääristä. Jos kahdella naapurikunnalla on eri määrä naapureita, niin kuntien välisen klikin potentiaalifunktio tuottaa näiden kuntien ehdollisiin todennäköisyyksiin eri arvot, vaikka asetelma on sama. Määritelmän 3.6 mukaan potentiaalifunktion arvo ei saa vaihdella näin. Esitetty menetelmä ei välttämättä tuota eheää Markovin satunnaiskenttää, joten naapureiden lukumäärän huomioiminen jätettiin murresanamallissa pois. Kuitenkin, jos Markovin satunnaiskentän ehdolliset jakaumat ovat normaalisia, niin naapurien lukumäärää saa silloin käyttää tietyin ehdoin [Bes75].

Kuvassa 6.9(a) on histogrammi tarkkuusmuuttujien $\boldsymbol{t}$ reunajakaumien odotusarvoista, kun käytettiin kattavuuden kolme paikallista simulaatiota. Koska tarkkuusmuuttujilla on pieni varianssi, niin odotusarvo kuvaa jakaumaa hyvin. Kuvaan 6.9(b) on piirretty kaikki kunnat niistä alun perin kerätyn sanamäärän ja vastaavan tarkkuus-

muuttujan t reunajakauman odotusarvon mukaan. Näillä kahdella tekijällä on suuri korrelaatio keskenään (laskettu arvo 0,975), vaikka kunnasta kerättyä sanamäärää ei erikseen käytetä mallissa. Jos murremallin oletukset ja kattavuuden kolme paikallisen simulaation tuottamat tulokset ovat hyväksyttäviä, niin silloin kerättyjen murresanojen määrien vaihtelu selittyy sillä, kuinka kattavasti kunta on tutkittu. Jos kunnasta on kerätty vain vähän sanoja, niin syynä on ennemmin tutkimuksen puute kuin se, ettei aineiston murresanoja käytettäisi kunnassa.



(a) Histogrammi tarkkuusmuuttujien $\boldsymbol{t}$ reunajakaumien odotusarvoille.

(b) Kunnissa havaittujen sanojen lukumäärien yhteys muuttujien $\boldsymbol{t}$ odotusarvoihin.

Kuva 6.9: Kuvat tarkkuusmuuttujien $\boldsymbol{t}$ yhteydestä kunnissa havaittujen sanojen lukumääriin. Tulokset laskettiin kattavuuden kolme paikallisesta simulaatiosta.

Testeissä kokeiltiin myös esimerkin 3.9 mukaista Markovin satunnaiskenttää, jossa yhden kokoisten klikkien arvot $\boldsymbol{\alpha}$ olivat muuttujia. Tarkoituksena oli kokeilla, mitä vaikutusta olisi sanojen esiintymistodennäköisyyksien kasvattamisessa tai laskemisessa koko Suomen laajuudessa. Jokaiselle sanalle asetettiin sanakohtaisen muuttujan β_s lisäksi muuttuja α_s, joka oli yhteinen kaikille yhden kokoisille klikeille. Tulokset eivät olleet toivottavia: muuttujat $\boldsymbol{\alpha}$ taipuivat suosimaan liikaa sanojen esiintymättömyyttä, kuten oli suurimmassa osassa Suomen kunnissa. Kuntakohtaiset sanojen käytön todennäköisyydet olivat MCMC-simulaation jälkeen hyvin lähellä joko ykköstä tai nollaa, ja suurin osa ykkösistä oli alkuperäisiä havaintoja.

Muuttujien $\boldsymbol{\beta}$ priorijakaumiksi asetettiin myös tasaisia jakaumia väliltä $[-10,\ +10]$. Kolmen kokoisessa paikallisessa simulaatiossa 17 100:sta β-muuttujasta yhteensä 54 sai reunajakauman odotusarvoksi negatiivisen arvon, joista 7 muuttujan odotusarvo oli itseisarvoltaan suurempi kuin 0,5. Suuri negatiivinen arvo tarkoittaa, että jos sana esiintyy jossakin kunnassa, niin se ei luultavasti esiinny naapurikunnissa. Negatiiviset arvot osuivat niiden murresanojen kohdalle, joiden havainnot muodostavat kartalla epätyypillisen kuvion. Negatiiviset odotusarvot ovat luultavasti seurausta

mallin lievästä ylisovittumisesta, eikä $\boldsymbol{\beta}$-muuttujien negatiivisia arvoja otettu mukaan lopulliseen malliin. Tarkkuusmuuttujille $\boldsymbol{t}$ kokeiltiin maltillisia Beta-jakaumia prioreina, mutta ne eivät vaikuttaneet tuloksiin.

Eräässä testissä kunnat jaettiin 14 tutkimusasteluokkaan sen mukaan, paljonko kunnissa oli sanahavaintoja. Kuntakohtaisen tarkkuusmuuttujien $\boldsymbol{t}$ sijaan käytettiin yhtä tarkkuusmuuttujaa jokaiselle tutkimusasteluokalle, mutta tuloksissa ei näkynyt merkittäviä eroja. Tästä voidaan päätellä, että malli ei ylisovitu tarkkuusmuuttujien kohdalla.

Paikallisen simulaation kattavuuden valinta perustui testeihin eri arvoilla. Kattavuuden kolme simulaation tulokset poikkesivat kattavuudesta kaksi selvästi, mutta kattavuuteen neljä ei ollut juuri eroa. Tehokkuussyistä kattavuudeksi valittiin kolme. Myös kaikille sanoille yhteistä voimakkuusmuuttujaa β kokeiltiin: tulokset vaihtelivat hyvästä erittäin huonoon sen mukaan, paljonko sanasta oli havaintoja ja esiintyivätkö ne lähekkäin toisiaan.

6.2.5 Mallin tarkentaminen

Murresanoille käytetty malli on yksinkertainen, eikä se ota huomioon läheskään kaikkia ilmiöön vaikuttavia tekijöitä. Seuraavaksi esitellään parannusehdotuksia mallille.

Murresana-aineiston kuntien naapuriverkon valinnassa voisi ottaa paremmin huomioon todellisen riippuvuusalueen, jolloin naapurustoon kuuluisi muitakin kuntia kuin rajanaapureita. Hyvä liikenneyhteys kahden kunnan välillä voi aiheuttaa riippuvuutta kuntien murresanoissa, vaikka kunnilla ei olisi yhteistä rajaa. Rannikkokuntien välillä voi olla vilkasta vesiliikennettä, vaikka kunnilla ei olisikaan yhteistä maarajaa. Nämä lisäykset vaativat kuitenkin asiantuntijatietoa kunnista. Naapurikunniksi voisi määrätä rajanaapureiden sijaan ne kunnat, joiden keskipisteiden väliset maantieteelliset etäisyydet ovat riittävän pieniä.

Esitelty murresanamalli ei ota huomioon naapurien lukumäärää. Jos kunnalla on vain yksi naapuri, niin kunta on oletettavasti tavallista riippuvaisempi ainoasta naapuristaan. Silloin näiden kahden kunnan välisen klikin potentiaalifunktionkin tulisi ehkä saada tavallista suurempia arvoja. Toisaalta riippuvuus kasvaisi myös toiseen suuntaan, mikä ei ole aina toivottavaa. Ylipäätään on mahdollista käyttää sanakohtaisten muuttujien $\boldsymbol{\beta}$ asemesta sana- ja klikkikohtaisia muuttujia, jolloin jokaisen

klikin erityisominaisuudet voitaisiin ottaa huomioon. Kuntien väliset liikenneyhteydet, asutusten yhtenäisyys ja kuntien yhteisen rajan pituus ovat kaikki esimerkkejä klikkien erityisominaisuuksista. Näiden huomioiminen saattaa kuitenkin monimutkaistaa mallia liiaksi.

Murresanamalliin on mahdollista lisätä uusia muuttujia, joiden avulla voidaan selvittää riippuvuuksia murresanojen esiintymisen ja eri tekijöiden välillä. Esimerkiksi, jos malliin lisätään datana kuntien kaupunkimaisuusaste, niin sanakohtaisella muuttujalla voidaan tutkia, esiintyykö sana todennäköisemmin kaupunkimaisissa kunnissa kuin muualla. Tällä tavalla ilmiölle voidaan löytää selittäviä tekijöitä ja arvioida niiden vaikutuksen voimakkuutta.

Murresanamallin muuttujien $\boldsymbol{\beta}$ ja $\boldsymbol{t}$ hyväksymistodennäköisyyksiä Markovin ketjussa voi parantaa kehittämällä parempia ehdotusjakaumia. Muuttujien reunajakaumat antavat vihjeitä sopivien jakaumien keksimiseen. Hyvä valinta parantaisi hyväksymistodennäköisyyksiä ja ketjun sekoittumista, jolloin tulokset tarkentuisivat.

Paikallisessa simulaatiossa annettiin datana kaikkien muuttujien arvot, joista ei ollut enintään kolmen mittaista polkua naapuriverkossa kyseisen sanan positiiviseen havaintoon. Paikallisen simulaatio kattavuus voisi olla vakioarvon sijaan sanakohtainen, jolloin kaikkien epäoleellisten muuttujien (ja vain niiden) arvot annettaisiin datana. Toisaalta kattavuus voisi vaihdella eri osissa Suomea: Lounais-Suomen pienten kuntien kohdalla voisi käyttää suurempaa kattavuutta. Paikallinen simulaatio voisi myös perustua kuntien keskipisteiden väliseen maantieteelliseen etäisyyteen.

Murresanojen välisten riippuvuuksien selvittäminen voisi olla kiinnostava tutkimuksen kohde. Sanoja on kuitenkin niin paljon, että kattava analyysi voi olla laskennallisesti mahdotonta.

6.3 Lintujen pesimäaineisto

Seuraavaksi hyödynnetään Markovin satunnaiskenttiä ja hierarkkista mallintamista Suomen lintujen pesimäaineiston tutkimisessa. Ensin tutustutaan aineistoon ja sitten esitellään sille kehitetty malli. Sitten käsitellään MCMC-simulaatiosta saatuja tuloksia lintujen pesinnälle ja ehdotetaan parannuksia malliin.

6.3.1 Aineiston kuvaus

Suomen lintujen pesimäaineisto sisältää 248 linnun aluekohtaiset pesimävarmuudet eri puolilla Suomea [VLK98]. Aineisto on syntynyt osana laajempaa projektia, jossa on tarkoitus selvittää myös eri lajien runsaus ja kannanvaihtelut Suomessa. Lintujen pesimätietoja on tähän mennessä kerätty kahteen otteeseen 1970- ja 1980-luvuilla, ja kolmas selvitys on käynnissä [Atl06].

Eläinlajien levinneisyyden tutkimisessa käytetään usein *atlaskartoitusta* [Hög95], jossa tutkittava maasto jaetaan sopivan kokoisiin neliömäisiin alueisiin, *atlasruutuihin*. Atlasruutuihin lähetetään tarkkailijoita, jotka merkitsevät muistiin havaintonsa kyseisestä ruudusta. Jos atlasruudut ovat pieniä, tutkimus vaatii paljon resursseja; suurista atlasruuduista taas ei voi tuottaa kovin tarkkaa aineistoa. Kompromissina on usein päädytty atlasruudun kokoon 10 km × 10 km.

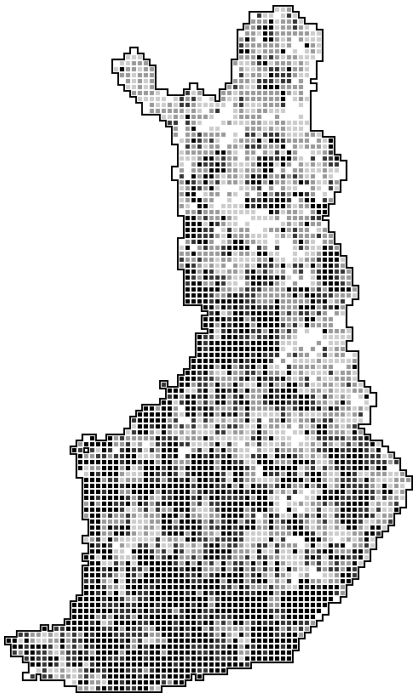
Lintujen pesimäaineistossa Suomi on jaettu yhtenäiskoordinaatiston avulla 3813 atlasruutuun, joista jokainen on kooltaan 10 km × 10 km. Projektin aluevastaavat ohjeistivat tuhansia lintuharrastajia ja lähettivät harrastajia eri atlasruutuihin. Yhteensä 248 linnun pesimätiedot kerättiin pitkälti vapaaehtoisten lintuharrastajien voimin. Aineisto sisältää *havaintoarvon* jokaisen linnun pesinnälle jokaisessa ruudussa. Havaintoarvot luokitellaan seuraavasti:

havainto	∼ merkitys
0	∼ epätodennäköinen pesintä (lintu läpikulkumatkalla tai ei havaintoa),
1	∼ mahdollinen pesintä (lintu sopivassa pesimäympäristössä),
2	∼ todennäköinen pesintä (soidinmenoja, pesänrakennusta),
3	∼ varma pesintä (poikasten ääniä, näköhavainto).

Pesimäaineiston laatu on yleensä ottaen todettu verrattain hyväksi. Silti joissakin atlasruuduissa on käynyt vain satunnaisia tarkkailijoita, joten erityisesti näistä ruuduista kerätty data on puutteellista [VLK98]. Havaintoarvojen varmuutta kuvataan seuraavalla atlasruutujen *selvitysasteluokittelulla*:

selvitysaste	∼ merkitys
1	∼ satunnaisia tarkkailijoita ruudussa,
2	∼ yleisimpiä maastotyyppejä on tutkittu,
3	∼ yleisimmät maastotyypit ja hyvät lintupaikat tutkittu,
4	∼ kaikki maastotyypit ja lintupaikat tutkittu (puolet koko ruudusta),
5	∼ eri maastotyyppejä ja lintupaikkoja tarkkailtu läpi pesimäajan.

Aineiston atlasruutujen selvitysasteet on esitetty kuvassa 6.10. Jatkossa atlasruutuja kutsutaan yksinkertaisesti ruuduiksi.



Kuva 6.10: Suomen atlasruutujen selvitysasteet: 1 (valkoinen), 2, 3, 4 ja 5 (musta).

6.3.2 Mallin kuvaus

Muodostetaan seuraavaksi lintujen pesintää kuvaava yksinkertainen malli. Malli on hyvin samankaltainen erään lintujen pesimäaineistolle aiemmin sovelletun mallin kanssa [HH94], eikä se poikkea murresana-aineistolle kehitetystä mallistakaan kovin paljon. Pesimäaineisto on yleensä ottaen murresana-aineistoa tarkempi. Pesimäaineistossa on silti selvästi puutteita, joten puuttuvaa dataa kannattaa mallintaa. Tavoitteena on selvittää ruutukohtaisten havaintoarvojen perusteella todennäköisyydet lintujen pesinnälle.

Ruudut luokitellaan niiden selvitysasteen $1, \ldots, 5$ mukaisesti viiteen *selvitysluokkaan* eli ruutujen joukkoon $R_1, \ldots, R_5$. Selvitysluokka R_i sisältää siis kaikki ne ruudut, joiden selvitysaste on i. Minkä tahansa selvitysluokan R_i ruutuja voidaan pitää yhtä hyvin tutkittuina.

Esitellään lyhyesti mallin sisältämät muuttujat. Datamatriisin $\boldsymbol{Y}$ alkio $y_{r,l} \in \{0,\ 1,\ 2,\ 3\}$ sisältää havainnon linnun l pesinnästä ruudussa r; binäärisen matriisin $\boldsymbol{X}$ alkio $x_{r,l}$ kertoo todellisen pesimätilanteen. Vektorin $\boldsymbol{\beta}$ reaaliarvoiset lintukohtaiset muuttujat säätävät naapuriruutujen välisen riippuvuuden voimakkuutta eli niiden saman-

kaltaisuutta. Reaalinen muuttuja $\lambda^{(0)}$ säätää nollasta poikkeavan havainnon todennäköisyyttä missä tahansa ruudussa, kun lintu ei todellisuudessa pesi siellä. Kun lintu pesii ruudussa, niin havaintoarvojen todennäköisyydet riippuvat vektorin $\boldsymbol{\lambda}^{(1)}$ selvitysluokkakohtaisista reaalisista muuttujista.

Jakauman katkaistuPoisson[6] avulla voidaan muodostaa lintujen pesimämallin riippuvuudet seuraavasti:

$$\begin{aligned}
[\beta_l] &\sim \text{Tas}(0,\ 10),\\
[x_{r,l} \mid \beta_l,\ \mathcal{N}(x_{r,l})] &\sim \text{Bernoulli}(p_{r,l}),\ \text{jossa logit}(p_{r,l}) = \beta_l \cdot \sum_{x_{u,l} \in \mathcal{N}(x_{r,l})} (2 \cdot x_{u,l} - 1),\\
[\lambda^{(0)}] &\sim \text{Tas}(0,\ 10),\\
[\lambda_R^{(1)}] &\sim \text{Tas}(0,\ 10),\\
[y_{r,l} \mid x_{r,l} = 0,\ \lambda^{(0)}] &\sim \text{katkaistuPoisson}_{\{0,1,2\}}(\lambda^{(0)}),\\
[y_{r,l} \mid x_{r,l} = 1,\ r \in R,\ \lambda_R^{(1)}] &\sim \text{katkaistuPoisson}_{\{0,1,2,3\}}(\lambda_R^{(1)}).
\end{aligned} \tag{6.11}$$

Lintumalli sisältää jokaisen linnun pesinnälle oman Markovin satunnaiskentän. Isingmalli sopii kuvaamaan lintujen pesintää, sillä lintujen voidaan olettaa leviävän enimmäkseen lähialueiden kautta. Ruuduille asetetaan naapureiksi kaikki (enintään) 8 ympäröivää ruutua. Naapuruston koko 8 soveltuu oletettavasti hieman paremmin kuin koko 4 [HH94]. Jotkut lintulajit viihtyvät suurissa parvissa enemmän kuin toiset, joten naapuriruutujen samankaltaisuuden voimakkuudet $\boldsymbol{\beta}$ asetetaan lintukohtaisiksi. Seuraavaksi käydään tarkemmin läpi ne muuttujat, joita ei esitelty murremallin yhteydessä.

Kun lintu ei todellisuudessa pesi ruudussa, niin havaintoarvojen todennäköisyydet riippuvat *valehavaintomuuttujasta* $\lambda^{(0)}$. Koska lintu ei pesi, havaintoarvo 3 (varma pesintä) on järkevää hylätä. Havaintoarvot 1 ja 2 ovat silti mahdollisia, sillä ne eivät tarkoita varmaa pesintää. Yksinkertaisuuden vuoksi muuttuja $\lambda^{(0)}$ oletetaan kaikille ruuduille ja linnuille yhteiseksi, ja riippuvuus ilmaistaan katkaistulla Poisson-jakaumalla. Muuttujan $\lambda^{(0)}$ priorijakaumaksi asetetaan epäinformatiivinen ja laskennallisesti tehokas tasainen jakauma väliltä $[0,\ 10]$ (kattaa tärkeimmät arvot).

[6] Jakauman $\text{katkaistuPoisson}_P(\lambda)$ määritelmä diskreetille pistejoukolle P on

$$\text{katkaistuPoisson}_P(k \mid \lambda) = \frac{1_{k \in P} \cdot \text{Poisson}(k \mid \lambda)}{\sum_{k' \in P} \text{Poisson}(k' \mid \lambda)},\ \text{jossa } 1_{ehto} = \begin{cases} 1 & \text{, jos } \textit{ehto} \text{ toteutuu} \\ 0 & \text{muuten.} \end{cases}$$

Kun lintu todella pesii ruudussa, niin havaintoarvojen 0, 1, 2 ja 3 todennäköisyydet saadaan katkaistusta Poisson-jakaumasta. Jakauman parametrina on vektorin $\boldsymbol{\lambda}^{(1)}$ se muuttuja, joka vastaa ruudun selvitysastetta. Näitä muuttujia nimitetään *havaintotarkkuusmuuttujiksi*, ja ne ovat yhteisiä kaikille linnuille ja saman selvitysluokan ruuduille. Havaintojen oletetaan siis jakautuneen samalla tavalla kaikissa samaan selvitysluokkaan kuuluvissa ruuduissa. Todellisuudessa lintujen havaintojen jakaumat poikkeavat lajeittain, mutta koska pesimäaineisto ei sisällä asiantuntijatietoa lintujen pesinnän havaitsemisen helppoudesta, lintujen havaintoarvojen oletetaan mallissa jakaantuneen samalla tavalla. Lintujen havaintojen erilaisuutta on jo pyritty vähentämään aineiston keräysvaiheessa, sillä kaikki maastotyypit pyrittiin tutkimaan. Havaintotarkkuusmuuttujien priorijakaumksi asetetaan tasaiset jakaumat väliltä $[0,\ 10]$.

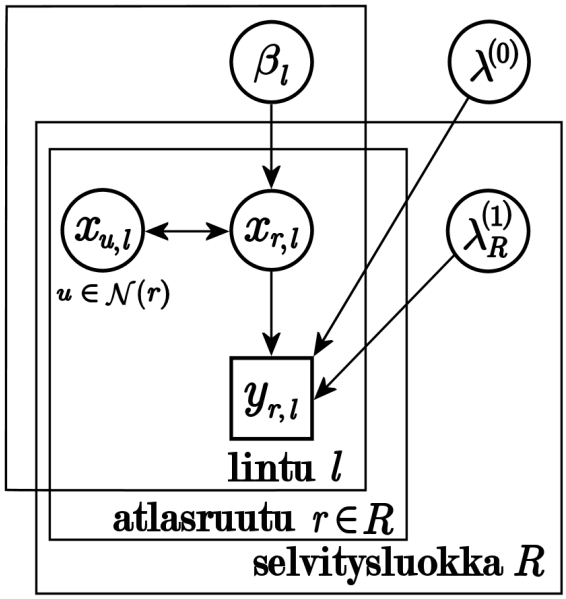
Havainto $y_{r,l}$ on ehdollisesti riippumaton linnun l pesinnästä muissa ruuduissa, kun ruudun r pesimätieto $x_{r,l}$ on annettu. Havainnon $y_{r,l}$ priorijakaumaksi valitaan katkaistu Poisson-jakauma. Perusteena on, että useiden lintujen pesinnästä saatujen havaintoarvojen 0, 1, 2 ja 3 jakaumat [Hög95, s. 73] näyttävät olevan approksimoitavissa katkaistulla Poisson-jakaumalla.

Lintujen pesimämalli on esitetty kuvassa 6.13. Kun merkitään $\boldsymbol{x_l} = (x_{1,l},\ x_{2,l},\ \ldots)$ muuttujia, jotka kuuluvat linnun l Markovin satunnaiskenttään, niin lintujen pesimämallin yhteisjakauma on

$$\begin{aligned}
&\textsc{pr}(\lambda^{(0)}, \boldsymbol{\lambda}^{(1)}, \boldsymbol{\beta}, \boldsymbol{X}, \boldsymbol{Y}) \\
&= \textsc{pr}(\lambda^{(0)}) \cdot \textsc{pr}(\boldsymbol{\lambda}^{(1)}) \cdot \textsc{pr}(\boldsymbol{\beta}) \cdot \textsc{Pr}(\boldsymbol{X} \mid \boldsymbol{\beta}) \cdot \textsc{Pr}(\boldsymbol{Y} \mid \boldsymbol{X},\ \lambda^{(0)},\ \boldsymbol{\lambda}^{(1)}) \\
&= \textsc{pr}(\lambda^{(0)}) \cdot \left(\prod_R \textsc{pr}(\lambda_R^{(1)}) \right) \cdot \prod_l \textsc{pr}(\beta_l) \cdot \textsc{Pr}(\boldsymbol{x_l} \mid \beta_l) \cdot \prod_R \prod_{r \in R} \textsc{Pr}(y_{r,l} \mid x_{r,l},\ \lambda^{(0)},\ \lambda_R^{(1)}).
\end{aligned} \tag{6.12}$$

6.3.3 Tulokset

Lintumallille suoritettiin MCMC-simulaatio, joka tuotti mallin posteriorijakauman reunajakaumista odotusarvot kaikille muuttujille. Erityisesti binääristen pesimämuuttujien $\boldsymbol{X}$ reunajakaumien odotusarvot tulkitaan pesimätodennäköisyyksiksi. Markovin ketjun pituus oli 110 000, josta lämmittelyvaihetta oli 10 000. Ohennusparametriksi valittiin 10. Simulaation kesto oli 5 tuntia (laitteiston nopeus 3 GHz).



Kuva 6.13: Hierarkkinen malli Suomen lintujen pesinnälle.

Konvergoitumistestit olivat samat kuin murresanamallissa – yksityiskohdat jätetään tällä kertaa väliin. Lyhyesti sanottuna konvergoitumisanalyysit antoivat ymmärtää, että ketjun konvergoitumista ei voi kiistää, kun lämmittelyvaiheen pituus on 1 000.

Naapuriruutujen samankaltaisuuden voimakkuutta säätävät muuttujat $\boldsymbol{\beta}$ päivitettiin Metropolis–Hastings-algoritmilla. Jokaisen muuttujan β_l ehdotusjakauma oli satunnaiskulku Normaali(b, $0{,}15^2$), jossa b on muuttujan β_l nykyinen arvo. Valinta tuotti suurimmalle osalle (90 %) vektorin $\boldsymbol{\beta}$ muuttujista hyväksymistodennäköisyyksiä väliltä 8–43 %. Päivityksen yhteydessä uskottavuustodennäköisyyttä approksimoitiin valeuskottavuudella.

Valehavaintomuuttujaa $\lambda^{(0)}$ päivitettiin Metropolis–Hastings-algoritmilla. Muuttujan reunajakauma on erittäin huipukas, joten satunnaiskulkua käyttävälle ehdotusjakaumalle saadaan suuria hyväksymistodennäköisyyksiä vain lyhyillä siirtymillä. Ehdotusjakaumaksi valittiin satunnaiskulku Normaali(c, $0{,}008^2$), jossa c on muuttujan $\lambda^{(0)}$ nykyinen arvo. Muuttujan hyväksymistodennäköisyys oli lopulta 8 %.

Myös havaintotarkkuusmuuttujien $\boldsymbol{\lambda}^{(1)}$ päivityksessä sovellettiin Metropolis–Hastings-algoritmia. Yksittäisen muuttujan $\lambda_R^{(1)}$ ehdotusjakaumaksi valittiin satunnaiskulku Normaali(c, $0{,}15^2$) (c on muuttujan $\lambda_R^{(1)}$ nykyinen arvo), mikä tuotti selvitysluokkia $R_1, \ldots, R_5$ vastaaville muuttujille hyväksymistodennäköisyydet 6 %, 7 %, 11 %, 19 % ja 24%.

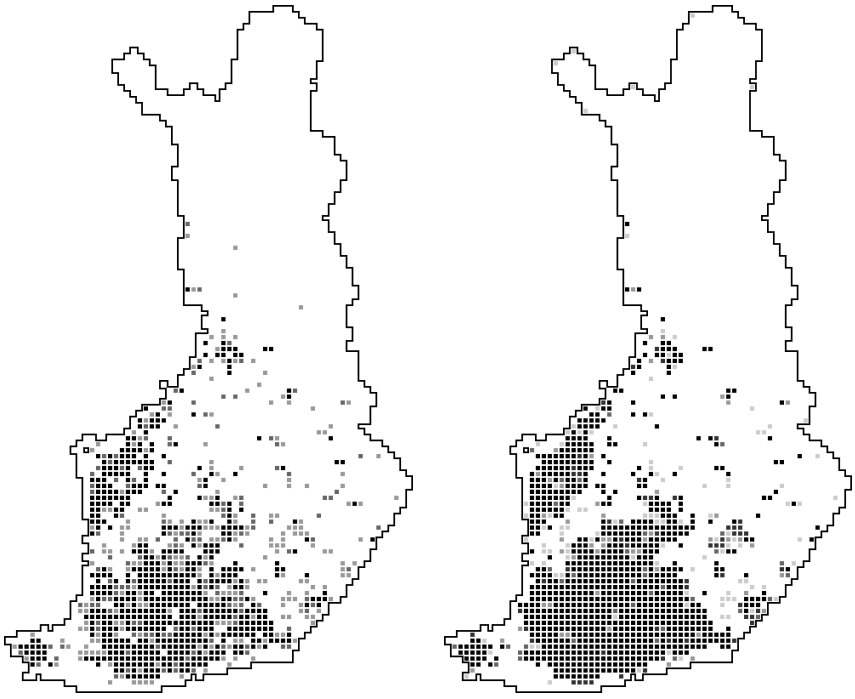
Pesimämuuttujat $\boldsymbol{X}$ päivitettiin Gibbsin algoritmilla. Markovin ketjun yhden siirtymän aikana mallin jokaista muuttujaa päivitettiin kerran.

Kuva 6.14 sisältää naakasta ja lapinsirkusta tehdyt havainnot sekä MCMC-simulaation tuottamat pesimätodennäköisyydet molemmille. Jos jossakin kuvan ruudussa on havaintoarvo 1 ja ympäröivät havainnot ovat nollia, niin lintu ei luultavasti pesi ruudussa. Toisaalta havaintoarvo 2 on kuvien perusteella hyvä merkki pesinnälle yksinäänkin – erityisesti selvitysasteeltaan heikoissa ruuduissa. Malli ei huomioi lainkaan Norjan puoleista maastoa, vaikka lapinsirkku siellä varmasti pesiikin. Kuvan 6.14(d) antamat todennäköisyydet ovat silti odotusten mukaisia, ilman Norjan ruutujen mallintamistakin. Kuva 6.14(b) paljastaa, että Ising-mallin reunavääristymä on läsnä myös lintumallissa: naakalla on muutamassa Pohjois-Suomen ruudussa epärealistisen korkea pesimätodennäköisyys. Syyt reunavääristymälle ovat samat kuin murresanojen yhteydessä. Kaikkiaan tulokset ovat uskottavia, vaikka mallissa on monia yksinkertaistuksia.

Koska havaintotarkkuusmuuttujien $\boldsymbol{\lambda}^{(1)}$ reunajakaumat ovat huipukkaita, jakaumia voidaan kuvata niiden odotusarvoilla. Selvitysluokkia $R_1, \ldots, R_5$ vastaavien muuttujien $\lambda^{(1)}_{R_1}, \ldots, \lambda^{(1)}_{R_5}$ reunajakaumien odotusarvoiksi saatiin 0,653; 1,834; 3,123; 4,768 ja 7,466. Kun lintu todella pesii ruudussa, niin havaintotarkkuusmuuttujat tuottavat ruudun havainnoille havaintoasteesta riippuvia todennäköisyyksiä. Edellisistä odotusarvoista lasketut approksimaatiot selvitysluokkien ruuduista saatujen havaintoarvojen jakaumille ovat seuraavassa taulukossa.

		havaintoarvo y			
katkaistuPoisson$_{\{0,1,2,3\}}(y \mid \mathtt{E}[\lambda^{(1)}_R])$		0	1	2	3
	1	0,523	0,341	0,112	0,024
	2	0,180	0,331	0,303	0,186
selvitysluokka R	3	0,071	0,222	0,346	0,361
	4	0,028	0,135	0,323	0,513
	5	0,009	0,071	0,264	0,656

Jos lintu todella pesii selvitysasteen 5 mukaan tutkitussa ruudussa, niin taulukon mukaan linnusta saadaan havaintoarvoksi 2 tai 3 todennäköisyydellä 0,920. Jos selvitysaste onkin 1, niin pesivästä linnusta saadaan vain heikko havainto (0 tai 1) todennäköisyydellä 0,864. Tästä voidaan päätellä, että hyvin tutkituissa ruuduissa on saatu hyviä havaintoja lähes kaikista niissä pesivistä linnuista. Lisäksi alkuperäisen aineiston heikosti selvitetyistä ruuduista näyttäisi puuttuvan havaintoja.



(a) Naakan havainnot.

(b) Naakan pesimätodennäköisyydet.

(c) Lapinsirkun havainnot.

(d) Lapinsirkun pesimätodennäköisyydet.

Kuva 6.14: Havainnot ja todennäköisyydet naakan (lintu 217) ja lapinsirkun (lintu 241) pesinnälle. Valkoinen, vaaleanharmaa, tummanharmaa ja musta väri vastaavat havaintokuvissa (a) ja (c) havaintoarvoja 0, 1, 2 ja 3. Samat värit ilmaisevat pesimätodennäköisyyskuvissa (b) ja (d) todennäköisyyksiä väliltä $[0, 1]$. Naakan ja lapinsirkun naapuririippuvuutta säätävien muuttujien β_{217} ja β_{241} odotusarvot ovat 0,48 ja 0,71.

Mallin posteriorijakaumasta lasketun valehavaintomuuttujan $\lambda^{(0)}$ reunajakauman odotusarvoksi saatiin 0,018. Tämä kiinteä arvo tuottaisi havainnoille 0, 1 ja 2 todennäköisyydet 0,98; 0,018 ja 0,0002 (havainto 3 kiellettiin). Jos lintu ei pesi ruudussa, niin havaintoarvo on siis lähes varmasti 0. Tästä voidaan päätellä karkeasti, että suurin osa pesimäaineiston nollasta poikkeavista havainnoista ovat merkkejä pesinnästä. Tämä vahvistaa myös ennakkotiedon aineiston verrattain hyvästä laadusta.

Pesimätodennäköisyydet laskettiin myös mallilla, jossa ruuduilla oli 8 naapurin sijaan 4 naapuria. Tuloksissa havaittiin pieniä eroja: naapuruston koko 8 vaikutti tuottavan eräässä mielessä yhtenäisempiä pesimäalueita. Kun naapuruston koko oli 4, niin muuttujien $\boldsymbol{\beta}$ arvot olivat noin kaksinkertaisia naapuruston kokoon 8 verrattuna – vaikutus oli lähes sama. Koska lintujen todellinen pesimätilanne on tuntematon, naapuruston valinta jää intuition varaan.

Kaikille selvitysluokille yhteisen valehavaintomuuttujan $\lambda^{(0)}$ sijaan kokeiltiin myös selvitysluokkakohtaisia valehavaintomuuttujia $\boldsymbol{\lambda}^{(0)}$. Silloin selvitysluokkaan R kuuluvan ruudun havaintojen prioritodennäköisyydet laskettiin katkaistusta Poisson-jakaumasta, jonka parametrina oli $\lambda_R^{(0)}$. Selvitysluokkakohtaisten muuttujien $\boldsymbol{\lambda}^{(0)}$ huipukkaiden reunajakaumien odotusarvoiksi saatiin 0,000; 0,000; 0,007; 0,020 ja 0,055. Tästä seuraa, että selvitysluokkien 1 ja 2 ruuduista kerätyt nollista poikkeavat havainnot tarkoittavat varmaa pesintää, mikä ei tunnu uskottavalta. Selvitysluokkien 1 ja 2 ruuduista voi nimittäin intuition mukaan saada nollasta poikkeavia havaintoja, vaikka lintu ei pesi ruudussa. Epärealististen odotusarvojen syntyminen saattaa olla ylisovittamisen tulosta. Voi myös olla, että pesimättömästä ruudusta saatujen havaintojen jakaumaksi valittu katkaistu Poisson-jakauma ei välttämättä toimi kovin hyvin – tämä vaatisi lisätutkimusta. Tulokset eivät olleet kovin mielekkäitä, joten ajatus selvitysastekohtaisista muuttujista hylättiin.

6.3.4 Mallin tarkentaminen

Edellisessä luvussa esiteltiin yksinkertainen malli, joka tuottaa järkeviä lintujen pesimätodennäköisyyksiä. Lintuasiantuntijoiden ja ylimääräisen datan avulla on kuitenkin mahdollista kehittää mallia tarkemmaksi. Käydään seuraavaksi läpi joitakin ehdotuksia mallin parantamiseksi.

Esitetyssä mallissa oletettiin, että havaintoarvot jakautuvat katkaistun Poisson-jakauman mukaisesti silloin, kun lintu pesii ruudussa. Havaintojen jakauma ei vält-

tämättä ole Poisson-jakautunut. Sen tilalla voisi käyttää jotakin diskreettiä jakaumaa, joka sallii enemmän vaihtelua jakauman muotoon. Toisaalta linnuista saatujen havaintojen jakauma vaihtelee linnuittain [Hög95, s. 73], joten havaintotarkkuusmuuttujat voisi asettaa lintukohtaisiksi. Tällainen menettely voi kuitenkin monimutkaistaa mallia liiaksi ja kasvattaa ylisovittumisen riskiä. Valehavaintomuuttujan $\lambda^{(0)}$ vaihtaminen huolellisemmin valittuun muuttujaan voisi myös tuottaa parempia tuloksia.

Lähellä Suomen rajaa sijaitsevat Ruotsin, Norjan ja Venäjän maaston ruudut voisi myös ottaa mukaan malliin. Vielä parempi, jos näistä ruuduista on saatavilla samaa lintujen pesimädataa. Rajantakaisten maa-alueiden lisääminen malliin olisi realistista: linnut eivät välitä valtioiden rajoista.

Yksinkertainen malli ei ota kovin hyvin huomioon lintujen erityispiirteitä. Erityisesti kesykyyhkyn eli pulun tiedetään viihtyvän kaupunkimaisessa ympäristössä. Jos ruuduista olisi tietoa niiden selvitysasteen lisäksi myös kaupunkimaisuudesta, niin pulun erityisominaisuudet voitaisiin ottaa huomioon. Jotkut linnut viihtyvät erityisesti pohjoisessa, joten ruutujen pohjoisuuden voisi myös huomioida. Ylipäätään on mahdollista lisätä ruutuihin mitä tahansa dataa, jonka merkitystä lintujen pesintään halutaan tutkia. Näin voidaan löytää selittäviä tekijöitä lintujen pesinnälle.

Voisi olla mielenkiintoista hyödyntää asiantuntijoiden tarjoamaa tietoa eri lintulajien välisistä riippuvuuksista. Toisaalta mallin avulla voitaisiin vahvistaa vanhoja uskomuksia tai jopa löytää uusia riippuvuuksia lajien välillä. Kuten murresana-aineiston yhteydessä, laskennalliset ongelmat voivat olla tässäkin liian suuria.

6.4 Jatkokäsittely

Bayesiläistä hierarkkista mallintamista ja Markovin satunnaiskenttiä käytettiin mallintamaan paikkatietoaineistoista puuttuvaa dataa. Sekä murremallissa että lintumallissa käytettiin binäärisiä muuttujia $\boldsymbol{X}$ kuvaamaan esiintymiä. Esiintymät tarkoittivat näissä tapauksissa murresanojen käyttämistä kunnissa tai lintujen pesintää atlasruuduissa. Mallien MCMC-simulaatiot tuottivat muuttujien reunajakaumille odotusarvot. Koska muuttujat $\boldsymbol{X}$ ovat binäärisiä, niiden odotusarvot tulkittiin esiintymien todennäköisyyksiksi. Nämä todennäköisyydet ovat (toivottavasti) luotettavampia kuin alkuperäisen aineiston data, joten jatkoanalyysit kannattaa suunnata uusiin todennäköisyyksiä sisältäviin aineistoihin.

Data-analyysissä on kehitetty lukuisia algoritmeja binääriselle datalle. Näitä algoritmeja ei voi kuitenkaan suoraan soveltaa todennäköisyysarvoja sisältävälle aineistolle. Tähän esitetään kolme ratkaisua. Ensinnäkin todennäköisyysaineistosta on mahdollista luoda binäärisiä aineistoja niin, että jokaisen todennäköisyysarvon kohdalla arvotaan, tuleeko arvoksi ykkönen vai nolla. Tällaisia binäärisiä aineistoja on syytä luoda useita. Menetelmässä on kuitenkin heikkous: koska binääriset arvot arvotaan suoraan todennäköisyyksistä, niin aineiston itsekorrelaatio vähenee. Toinen vaihtoehto on luokitella todennäköisyysmatriisin arvot MPM-estimaatin binäärisiksi arvoiksi. Kolmas menetelmä tallentaa MCMC-simulaation aikana silloin tällöin muuttujien $\boldsymbol{X}$ arvot, jolloin jokainen arvoyhdistelmä vastaa binääristä aineistoa. Tämä menetelmä säilyttää luultavasti aineiston itsekorrelaation edellisiä menetelmiä paremmin. Joka tapauksessa, kaikilla menetelmillä tuotettuja binäärisiä aineistoja voi analysoida binääriselle datalle suunnatuilla algoritmeilla.

7 Yhteenveto

Tutkielmassa käsiteltiin paikkatietoaineistosta puuttuvan datan mallintamista. Jos alkuperäisessä aineistossa esiintyy epävarmuutta, niin jatkoanalyysinkin tulokset ovat epävarmoja. Paikkatietoaineistosta puuttuvaa dataa on mahdollista mallintaa, sillä esimerkiksi lähialueet ovat usein toistensa kaltaisia.

Markovin satunnaiskenttiä käytettiin mallintamaan lähialueiden välisiä riippuvuuksia. Niiden merkittävin laskennallinen ominaisuus on ehdollinen riippumattomuus: satunnaiskentän solmut ovat riippumattomia muista solmuista, kun naapurisolmut on annettu. Silti Markovin satunnaiskentät ovat tarpeeksi ilmaisuvoimaisia monimutkaistenkin riippuvuuksien esittämiseen. Monimutkaisten riippuvuuksien esittäminen on työlästä, mutta usein riittääkin mallintaa yksinkertainen riippuvuus. Markovin satunnaiskentät perustuvat todennäköisyyslaskentaan, joten niitä voi käyttää osana bayesiläisiä todennäköisyysmalleja.

Bayesiläisiä hierarkkisia malleja voi käyttää mallintamaan lähes mielivaltaisen monimutkaisia ilmiöitä. Ehtona on, että ilmiön riippuvuudet voidaan esittää todennäköisyyskäsitteen avulla. Myös tässä on taustalla laskennallinen tehokkuus: hierarkkisen mallin muuttujat riippuvat yleensä vain joistakin muista tarkasti valituista muuttujista.

Bayesiläisten mallien käsittely on joka tapauksessa laskennallisesti vaativaa. Jotta mallin posteriorijakaumaa voitaisiin approksimoida, on käytännössä pakko soveltaa Markovin ketju Monte Carlo -menetelmää (MCMC). MCMC-menetelmien toteuttamisessa on huomioitava lukuisia yksityiskohtia, jotta sen antamat tulokset olisivat oikeita. Tulosten oikeellisuus on syytä tarkastaa mahdollisimman monella analyysimenetelmällä. MCMC-menetelmän käyttäminen on aikaa vievää, mutta usein ainoa tapa approksimoida posteriorijakaumaa.

Tutkielmassa esiteltyjä menetelmiä sovellettiin kahteen paikkatietoaineistoon: suomen murresana-aineistoon ja Suomen lintujen pesimäaineistoon. Aineistoille kehitetyt mallit olivat yksinkertaisia ja sisälsivät erään Markovin satunnaiskentän, Ising-mallin. Vaikka mallit olivat yksinkertaisia, niin tulokset olivat varsin uskottavia. Merkittävimmät heikkoudet olivat ajan käyttö (vuorokausia) ja tuloksissa esiintynyt reunavääristymä: murresanojen käytön ja lintujen pesinnän todennäköisyydet vääristyvät reuna-alueilla. Murresanamallille sovellettiin myös uutta yksinkertaista menetelmää, paikallista simulaatiota, joka vähensi selvästi reunavääristymän tuottamien epärealististen todennäköisyyksien määrää.

Puuttuvan datan mallintaminen on laskennallisesti mahdollista myös suurille paikkatietoaineistoille, vaikka aikaa vievää. Tutkielmassa esitetyt tulokset rohkaisevat käyttämään esitettyjä menetelmiä myös muihin paikkatietoaineistoihin, joiden sisältämää epävarmuutta halutaan vähentää. Parhaassa tapauksessa epävarmuuden vähentäminen aineistosta poistaa epävarmuutta myös jatkoanalyysin tuloksista.

Lähteet

AH97 Arjas, E. ja Heikkinen, J., An Algorithm for Nonparametric Bayesian Estimation of a Poisson Intensity. *Computational Statistics*, 12,3(1997), sivut 385–402.

Atl06 Luonnontieteellisen keskusmuseon eläinmuseo: Suomen lintuatlas (www-sivu). URL `http://www.lintuatlas.fi/`. URL tarkastettu 11.10.2006.

Aur91 Aurenhammer, F., Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23,3(1991), sivut 345–405.

BCV95 Best, N., Cowles, M. K. ja Vines, K., CODA, Convergence Diagnosis and Output Analysis Software for Gibbs sampling output, version 0.30. Tekninen raportti, MRC Biostatistics Unit, Institute of Public Health, Cambridge CB2 2SR, elokuu 1995. URL `http://www.mrc-bsu.cam.ac.uk/bugs/documentation/contents.shtml`. URL tarkastettu 2.10.2006.

Bes74 Besag, J., Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society, series B*, 36,2(1974), sivut 192–236.

Bes75 Besag, J., Statistical Analysis of Non-Lattice Data. *The Statistician*, 24,3(1975), sivut 179–195.

BG98 Brooks, S. P. ja Gelman, A., General Methods for Monitoring Convergence of Iterative Simulations. *Journal of Computational and Graphical Statistics*, 7,4(1998), sivut 434–455. URL `http://www.amstat.org/publications/jcgs/pdf_98/Brooks.pdf`. URL tarkastettu 2.10.2006.

Bol98 Bollobás, B., *Modern Graph Theory*. Springer-Verlag, New York, Yhdysvallat, 1998. Luku I.4.

BRT05 Best, N., Richardson, S. ja Thomson, A., A Comparison of Bayesian Spatial Models for Disease Mapping. *Statistical Methods in Medical Research*, 14, sivut 35–59.

BYM91 Besag, J., York, J. ja Mollié, A., Bayesian Image Restoration with Two Applications in Spatial Statistics. *Annals of the Institute of Statistical Mathematics*, 43,1(1991), sivut 1–59.

CC96 Cowles, M. K. ja Carlin, B. P., Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *Journal of the American Statistical Association*, 91,434(1996), sivut 883–904.

DLR77 Dempster, A. P., Laird, N. M. ja Rubin, D. B., Maximum Likelihood from Incomplete Data via the *EM*-algorithm. *Journal of the Royal Statistical Society, Series B*, 39,1(1977), sivut 1–38.

Gam97 Gamerman, D., *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. Chapman-Hall, Iso-Britannia, 1997.

GCSR04 Gelman, A., Carlin, J. B., Stern, H. S. ja Rubin, D. B., *Bayesian Data Analysis*. Chapman-Hall, 2004. Toinen laitos.

GG84 Geman, S. ja Geman, D., Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–6,6(1984), sivut 721–741. URL `http://cis.jhu.edu/people/faculty/geman/publications.html`. URL tarkastettu 2.10.2006.

GLV00 Galluccio, A., Loebl, M. ja Vondrák, J., New Algorithm for the Ising Problem: Partition Function for Finite Lattice Graphs. *Physical Review Letters*, 84,26(2000), sivut 5924–5927. URL `http://research.microsoft.com/users/vondrak/data/ising.ps`. URL tarkastettu 2.10.2006.

Goo86 Goodchild, M. F., *Spatial Autocorrelation*. Numero 47 sarjassa Concepts and Techniques in Modern Geography (CATMOG). Geo Books, Norwich, Iso-Britannia, 1986. Luku 1.

GR92 Gelman, A. ja Rubin, D. B., Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7,4(1992), sivut 457–472.

Gre95 Green, P. J., Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination. *Biometrika*, 82,4(1995), sivut 711–732.

GRS96 Gilks, W. R., Richardson, S. ja Spiegelhalter, D. J., *Markov Chain Monte Carlo in Practice*. Chapman & Hall, Lontoo, Iso-Britannia, 1996. (toimitettu teos).

GS90 Gelfand, A. E. ja Smith, A. F. M., Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association*, 85,410(1990), sivut 398–409.

Has70 Hastings, W. K., Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57,1(1970), sivut 97–109.

Hec97 Heckerman, D., Bayesian Networks for Data Mining. *Data Mining and Knowledge Discovery*, 1,1(1997), sivut 79–119. URL `ftp://ftp.`

research.microsoft.com/pub/tr/tr-95-06.pdf. Artikkeli on julkaistu useilla eri otsikoilla. URL tarkastettu 2.10.2006.

HH94 Heikkinen, J. ja Högmander, H., Fully Bayesian Approach to Image Restoration with an Application in Biogeography. *Applied Statistics*, 43,4(1994), sivut 569–582.

Hög95 Högmander, H., *Methods of Spatial Statistics in Monitoring of Wildlife Populations*. Väitöskirja, Jyväskylän yliopisto, 1995.

KAV05 Kasetkasem, T., Arora, M. K. ja Varshney, P. K., Super-Resolution Land Cover Mapping Using a Markov Random Field Based Approach. *Remote Sensing of Environment*, 96, sivut 302–314.

KGV83 Kirkpatrick, S., Gelatt, Jr., C. D. ja Vecchi, M. P., Optimization by Simulated Annealing. *Science*, 220,4598(1983), sivut 671–680.

KS80 Kindermann, R. ja Snell, J. L., *Markov Random Fields and their Applications*. American Mathematical Society, Rhode Island, Yhdysvallat, 1980. URL http://www.ams.org/online_bks/conm1/conm1-whole.pdf. Luvut 1 ja 2. URL tarkastettu 2.10.2006.

Li01 Li, S. Z., *Markov Random Field Modeling in Image Analysis*. Springer-Verlag, Tokio, Japani, 2001. Luvut 1, 2 ja 3.

MGM06 Murray, I., Ghahramani, Z. ja MacKay, D. J. C., MCMC for doubly-intractable distributions. *In Proceedings of the 22nd Conference of Uncertainty in Artificial Intelligence 2006*, Cambridge, Massachusetts, Yhdysvallat, heinäkuu 2006, sivut 359–366, URL http://www.gatsby.ucl.ac.uk/~iam23/pub/06doubly_intractable/doubly_intractable.pdf. URL tarkastettu 2.10.2006.

MRR+53 Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. ja Teller, E., Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21,6(1953), sivut 1087–1092.

MSTU02 Myllymäki, P., Silander, T., Tirri, H. ja Uronen, P., B-Course: A Web-Based Tool for Bayesian and Causal Data Analysis. *International Journal on Artificial Intelligence Tools*, 11,3(2002), sivut 369–387.

URL http://cosco.hiit.fi/Articles/ijait02.pdf. URL tarkastettu 2.10.2006.

Nea93 Neal, R. M., Probabilistic Inference Using Markov Chain Monte Carlo Methods. CRG-TR-93-1, Toronton yliopisto, syyskuu 1993. URL http://www.cs.toronto.edu/~radford/ftp/review.pdf. URL tarkastettu 2.10.2006.

Nea03 Neal, R. M., Slice Sampling. *The Annals of Statistics*, 31,3(2003), sivut 705–767.

Pre74 Preston, C. J., *Gibbs States on Countable Sets*. Cambridge University Press, Iso-Britannia, 1974. Luku 1.

PW96 Propp, J. G. ja Wilson, D. B., Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. *Random Structures and Algorithms*, 9,1–2(1996), sivut 223–252. URL http://www.math.wisc.edu/~propp/sample.ps.gz. URL tarkastettu 2.10.2006.

RC04 Robert, C. P. ja Casella, G., *Monte Carlo Statistical Methods*. Springer-Verlag, New York, Yhdysvallat, 2004. Toinen laitos.

RH05 Rue, H. ja Held, L., *Gaussian Markov Random Fields*. Numero 104 sarjassa Monographs on Statistics and Applied Probability. Chapman-Hall/CRC, Boca Raton, Yhdysvallat, 2005. URL http://www.statsnetbase.com/. URL tarkastettu 2.10.2006.

Smi05 Smith, B. J., Bayesian Output Analysis Program (BOA) Version 1.1 User's Manual. URL http://www.public-health.uiowa.edu/boa/BOA.pdf. URL tarkastettu 2.10.2006, tammikuu 2005.

SP94 Srinivas, M. ja Patnaik, L. M., Genetic Algorithms: a Survey. *Computer*, 27,6(1994), sivut 17–26.

STBG96 Spiegelhalter, D., Thomas, A., Best, N. ja Gilks, W., Bugs 0.5, Bayesian inference Using Gibbs Sampling, Manual (version ii). Tekninen raportti, MRC Biostatistics Unit, Institute of Public Health, Cambridge CB2 2SR, toukokuu 1996. URL http://www.mrc-bsu.cam.ac.uk/bugs/documentation/contents.shtml. URL tarkastettu 2.10.2006.

TB98 Tjelmeland, H. ja Besag, J., Markov Random Fields with Higher-order Interactions. *Scandinavian Journal of Statistics*, 25,3(1998), sivut 415–433.

Tie96 Tierney, L., Introduction to General State-Space Markov Chain Theory. Teoksessa [GRS96], luku 4, s. 59–74.

TMSL98 Toivonen, H., Mannila, H., Salmenkivi, M. ja Laakso, K.-P., Bassist – A Tool for MCMC Simulation of Statistical Models. *In Proceedings of the Eurosim '98 Simulation Congress*, Helsinki, Suomi, huhtikuu 1998, sivut 590–595, URL `http://citeseer.ist.psu.edu/toivonen98bassist.html`. URL tarkastettu 2.10.2006.

Tuo89 Tuomi, T., *Suomen murteiden sanakirja, johdanto.* Numero 36 sarjassa Kotimaisten kielten tutkimusjulkaisuja. Kotimaisten kielten tutkimuskeskus (KOTUS), Valtion painatuskeskus, Helsinki, 1989.

VLK98 Väisänen, R. A., Lammi, E. ja Koskimies, P., *Muuttuva pesimälinnusto.* Otava, Keuruu, 1998.

WH97 Wu, H. ja Huffer, F. W., Modelling the Distribution of Plant Species Using the Autologistic Regression Model. *Environmental and Ecological Statistics*, 4,1(1997), sivut 49–64.

Win95 Winkler, G., *Image Analysis, Random Fields and Dynamic Monte Carlo Methods.* Springer-Verlag, Berliini, Saksa, 1995. Luvut 2 ja 3.

ZBS01 Zhang, Y., Brady, M. ja Smith, S., Segmentation of Brain MR Images Through a Hidden Markov Random Field Model and the Expectation-Maximization Algorithm. *IEEE Transactions on Medical Imaging*, 20,1(2001), sivut 45–57.

Liite 1. Ising-mallin ehdolliset todennäköisyydet

Tarkastellaan Ising-mallin todennäköisyyskuvausta

$$\Pr(\boldsymbol{a}) = \frac{1}{Z} \cdot \exp\left(\sum_{\{X_i\} \in \mathcal{C}_1} \alpha_i \cdot a_i + \sum_{\{X_i, X_j\} \in \mathcal{C}_2} \beta_{i,j} \cdot 1_{a_i = a_j} \right).$$

Johdetaan tästä ehdollinen todennäköisyys

$$\Pr(X_k = a_k \mid X_j = a_j,\ j \neq k) = (*)$$

$$(*) = \frac{\frac{1}{Z} \cdot \exp\left(\sum_{\{X_i\} \in \mathcal{C}_1} \alpha_i \cdot a_i + \sum_{\{X_i, X_j\} \in \mathcal{C}_2} \beta_{i,j} \cdot 1_{a_i = a_j} \right)}{\sum_{a_k \in \{0,1\}} \frac{1}{Z} \cdot \exp\left(\sum_{\{X_i\} \in \mathcal{C}_1} \alpha_i \cdot a_i + \sum_{\{X_i, X_j\} \in \mathcal{C}_2} \beta_{i,j} \cdot 1_{a_i = a_j} \right)}.$$

Edellisessä on vielä mukana kaikki klikit. Supistetaan ensin normalisointivakiot Z pois. Jaetaan klikit kahteen osaan sen mukaan, kuuluuko ruutu k niihin. Havaitaan että ne klikit, joihin ei kuulu ruutua k, voidaan supistaa pois. Jäljelle jää

$$(*) = \frac{\exp\left(\alpha_k \cdot a_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot 1_{a_k = a_j} \right)}{\sum_{a_k \in \{0,1\}} \exp\left(\alpha_k \cdot a_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot 1_{a_k = a_j} \right)}.$$

$$(*) = \frac{\exp\left(\alpha_k \cdot a_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot 1_{a_k = a_j} \right)}{\exp\left(\sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot (1 - a_j) \right) + \exp\left(\alpha_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot a_j \right)}$$

Supistetaan jakajan ensimmäisellä termillä ja sievennetään, niin saadaan

$$(*) = \frac{\exp\left(a_k \cdot (\alpha_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot (2 \cdot a_j - 1)) \right)}{1 + \exp\left(\alpha_k + \sum_{X_j \in \mathcal{N}(X_k)} \beta_{k,j} \cdot (2 \cdot a_j - 1) \right)}.$$

Kun ruudun k naapuriruutujen arvot tunnetaan, niin ehdolliset todennäköisyydet ruudun k arvoille saadaan sijoittamalla edelliseen lausekkeeseen $a_k = 0$ tai $a_k = 1$. Jos pätee $\beta_{k,j} = \beta$ kaikilla k ja j, niin saadaan yksinkertaisemmin

$$(*) = \frac{\exp\left(a_k \cdot (\alpha_k + \beta \cdot \sum_{X_j \in \mathcal{N}(X_k)} (2 \cdot a_j - 1)) \right)}{1 + \exp\left(\alpha_k + \beta \cdot \sum_{X_j \in \mathcal{N}(X_k)} (2 \cdot a_j - 1) \right)}.$$

Jos merkitään $p = \Pr(X_k = 1 \mid X_j = a_j,\ j \neq k)$, niin edellinen lauseke voidaan muotoilla seuraavasti:

$$\operatorname{logit}(p) = \ln\left(\frac{p}{1-p} \right) = \alpha_k + \beta \cdot \sum_{X_j \in \mathcal{N}(X_k)} (2 \cdot a_j - 1).$$

Ehdollisissa todennäköisyyksissä ei tarvitse laskea normalisointivakiota Z.

Liite 2. Markovin ketjun analyysi

Analyysit murresanamallin tarkkuusmuuttujalle t_{313} ja Markovin satunnaiskentän voimakkuusmuuttujalle β_{39}. Ensin testataan arvojen konvergoituminen Markovin ketjussa. Sitten testataan arvojen itsekorrelaatio ketjun sisällä ja vielä muuttujien reunajakaumien normaalisuus. Kaikki analyysit on tehty kolmen kokoisesta paikallisesta simulaatiosta saaduille tuloksille. Analysoinnissa käytettiin R-kielellä toteutettua BOA-pakettia.

Analyysitulokset muuttujalle t_{313}:

```
------------------------------------------------
BROOKS, GELMAN AND RUBIN CONVERGENCE DIAGNOSTICS:
------------------------------------------------
(Muuttujan t_313 hetkien 4 000 -- 17 900 arvoille ohennuksella 10.
Kaikilla neljällä ketjulla alkuarvoilla beeta=0.0 tai beeta=4.0 JA t=0.0 tai t=1.0)
Potential Scale Reduction Factor: 1.001263
Multivariate Potential Scale Reduction Factor = 1.002236
Corrected Scale Reduction Factors:  Estimate     0.975
                                    1.001368  1.004795
(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 4000.)

-----------------------------------------------------------------
HEIDLEBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS:
-----------------------------------------------------------------
(Muuttujan t_313 hetkien 0 -- 17 900 arvoille ohennuksella 10.)
Halfwidth test accuracy = 0.1

ALKUARVOT
beeta   t --> Stationarity  Keep  Discard  C-von-M  Halfwidth       Mean  Halfwidth
                      Test                              Test
0.0    0.0          passed  1432      358  0.3824808   passed  0.3998324  0.001076625
0.0    1.0          passed  1432      358  0.1772649   passed  0.4004784  0.0007564647
4.0    0.0          passed  1790        0  0.1984982   passed  0.3999095  0.0009672587
4.0    1.0          passed  1432      358  0.3930751   passed  0.4012423  0.0006631754

(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 3580.)

------------------------------
GEWEKE CONVERGENCE DIAGNOSTIC:
------------------------------
(Muuttujan t_313 hetkien 8 000 -- 17 900 arvoille ohennuksella 10.)
Fraction in first window = 0.1
Fraction in last window = 0.5

ALKUARVOT
beeta   t  -->     Z-Score     p-value
0.0    0.0     -0.02532771  0.97979357
0.0    1.0     -0.64747060  0.51732740
```

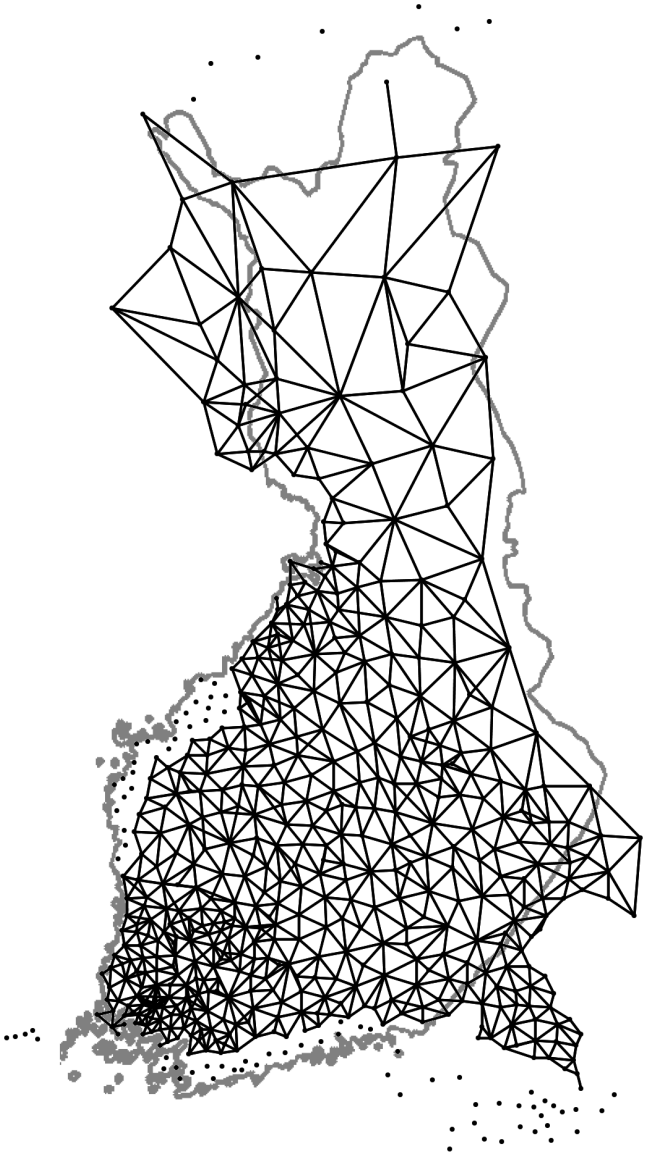
```
4.0    0.0     -1.24018030  0.21490870
4.0    1.0     -1.31975040  0.18691840

(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 8000.)

----------------------------------------
RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC:
----------------------------------------
(Muuttujan t_313 hetkien 0 -- 17 900 ohentamattomille arvoille)
Quantile = 0.025
Accuracy = +/- 0.005
Probability = 0.95

ALKUARVOT
beeta   t  --> Thin  Burn-in  Total  Lower Bound  Dependence Factor
0.0    0.0       10       40  57300         3746           15.29632
0.0    1.0        9       27  43479         3746           11.60678
4.0    0.0        8       32  48928         3746           13.06140
4.0    1.0        8       24  39952         3746           10.66524

(JOHTOPÄÄTÖS: Markovin ketjun tilojen väliset riippuvuudet ovat liian suuria, ja
riippuvuutta tulisi vähentää. Riippuvuuksista johtuen konvergoituminen on epävarmaa.
Simuloidun Markovin ketjun pituus 110 000 riittää ja ohennusparametri 10 on sopiva.)

--------------------------
LAGS AND AUTOCORRELATIONS:
--------------------------
(Muuttujan t_313 hetkien 10 000 -- 17 900 ohentamattomille arvoille)

ALKUARVOT
beeta   t  -->     Lag 1      Lag 5    Lag 10         Lag 50
0.0    0.0     0.8081397  0.3616903  0.1688950  0.0177048800
0.0    1.0     0.8132097  0.3814483  0.1592480  0.0364686900
4.0    0.0     0.8210105  0.4123207  0.1766053  0.0334045400
4.0    1.0     0.826228   0.3925320  0.1481468  0.0008347025

---------------------------------
NORMAALISUUSTESTI (SHAPIRO-WILK):
---------------------------------
(Normaalisuustestissä käytettiin muuttujan t_313 hetkien 10 000 -- 17 900 arvoja ohennuksella 30.
Testi on luotettavampi, kun arvot eivät korreloi keskenään kovin paljon.)

ALKUARVOT
beeta   t  -->      W  p-value
0.0    0.0     0.9936  0.33190
0.0    1.0     0.9896  0.05762
4.0    0.0     0.9939  0.36420
4.0    1.0     0.9920  0.16600

(JOHTOPÄÄTÖS: muuttujan t_313 reunajakauman normaalisuutta ei voi hylätä.)
```

Analyysitulokset muuttujalle β_{39}:

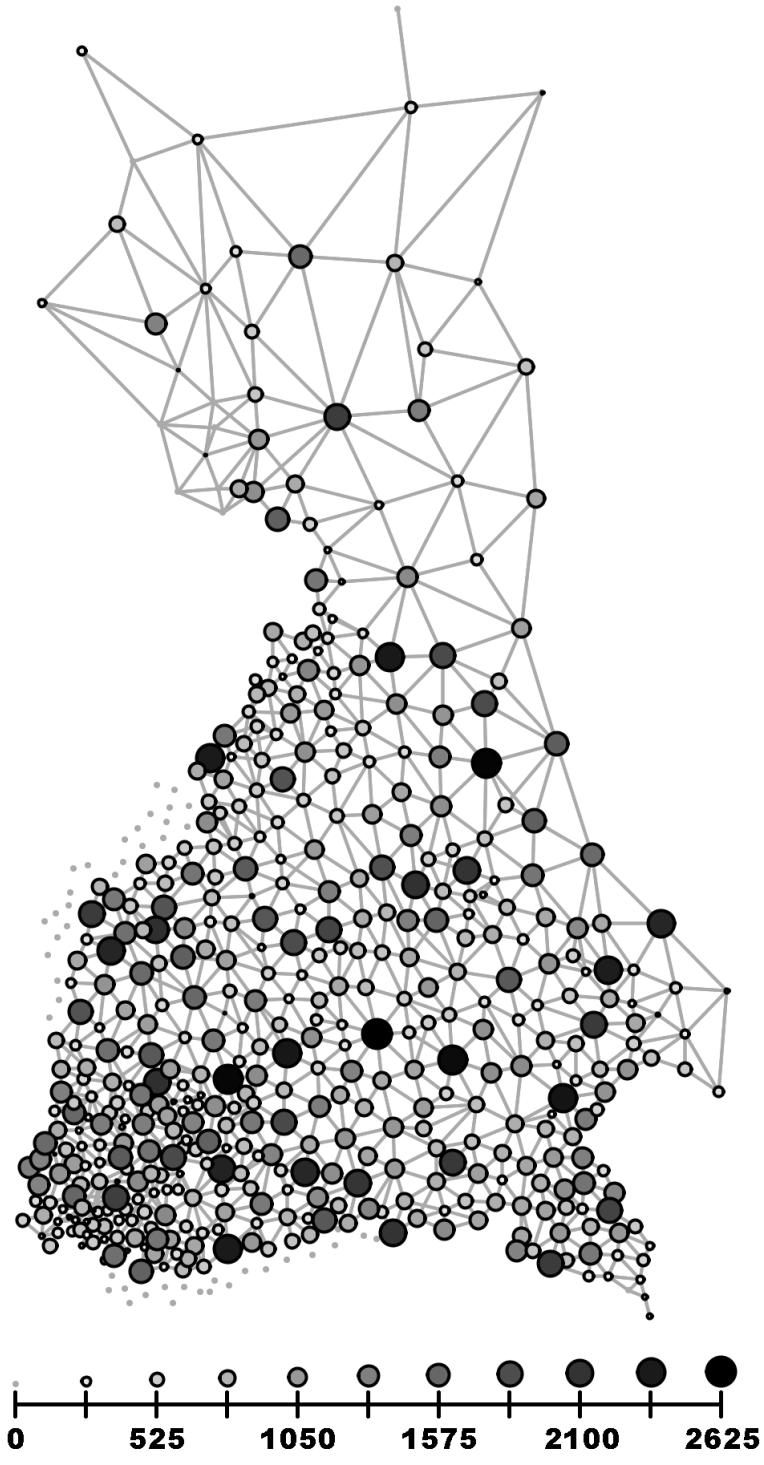
```
------------------------------------------------
BROOKS, GELMAN AND RUBIN CONVERGENCE DIAGNOSTICS:
------------------------------------------------
(Muuttujan beeta_39 hetkien 0 -- 17 900 arvoille ohennuksella 10.
Kaikilla neljällä ketjulla alkuarvoilla beeta=0.0 tai beeta=4.0 JA t=0.0 tai t=1.0)
Potential Scale Reduction Factor: 1.000209
Multivariate Potential Scale Reduction Factor = 1.000502
Corrected Scale Reduction Factors:  Estimate     0.975
                                    1.003827  1.004887
(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 0.)

---------------------------------------------------------------
HEIDLEBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS:
---------------------------------------------------------------
(Muuttujan beeta_39 hetkien 0 -- 17 900 arvoille ohennuksella 10.)
Halfwidth test accuracy = 0.1

ALKUARVOT
beeta   t --> Stationarity  Keep  Discard     C-von-M  Halfwidth      Mean   Halfwidth
                     Test                                 Test
0.0    0.0          passed  1790        0  0.39974040     passed  1.527056  0.03760006
0.0    1.0          passed  1790        0  0.08581159     passed  1.497190  0.02308078
4.0    0.0          passed  1790        0  0.14103400     passed  1.514115  0.02225621
4.0    1.0          passed  1790        0  0.12045040     passed  1.517011  0.02424531

(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 0.)

------------------------------
GEWEKE CONVERGENCE DIAGNOSTIC:
------------------------------
(Muuttujan beeta_39 hetkien 0 -- 17 900 arvoille ohennuksella 10.)
Fraction in first window = 0.1
Fraction in last window = 0.5

ALKUARVOT
beeta   t  -->     Z-Score     p-value
0.0    0.0      1.39836200  0.16200440
0.0    1.0      0.06925167  0.94478930
4.0    0.0     -0.15876960  0.87385040
4.0    1.0     -0.21339940  0.83101550

(JOHTOPÄÄTÖS: Konvergoitumista ei voi kiistää, kun lämmittelyvaiheen kesto on 0.)

----------------------------------------
RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC:
----------------------------------------
(Muuttujan beeta_39 hetkien 0 -- 17 900 ohentamattomille arvoille)
Quantile = 0.025
Accuracy = +/- 0.005
Probability = 0.95

ALKUARVOT
beeta   t  --> Thin  Burn-in  Total  Lower Bound  Dependence Factor
0.0    0.0       10       30 46800        3746          12.49333
```

```
0.0    1.0       11      33 40348        3746          10.77096
4.0    0.0        8      24 37768        3746          10.08222
4.0    1.0        9      36 38448        3746          10.26375

(JOHTOPÄÄTÖS: Markovin ketjun tilojen väliset riippuvuudet ovat liian suuria, ja
riippuvuutta tulisi vähentää. Riippuvuuksista johtuen konvergoituminen on epävarmaa.
Simuloidun Markovin ketjun pituus 110 000 riittää ja ohennusparametri 10 on sopiva.)

--------------------------
LAGS AND AUTOCORRELATIONS:
--------------------------
(Muuttujan t_313 hetkien 10 000 -- 17 900 ohentamattomille arvoille)

ALKUARVOT
beeta   t  -->     Lag 1      Lag 5      Lag 10        Lag 50
0.0    0.0     0.8479455  0.4593067  0.22203300   0.005852328
0.0    1.0     0.8139375  0.3522831  0.07143616  -0.008559286
4.0    0.0     0.8345370  0.4022273  0.16521080  -0.006775480
4.0    1.0     0.8036335  0.3534283  0.14922920   0.049917350

--------------------------------
NORMAALISUUSTESTI (SHAPIRO-WILK)
--------------------------------
(Normaalisuustestissä käytettiin muuttujan beeta_39 hetkien 10 000 -- 17 900 arvoja ohennuksella 30.
Testi on luotettavampi, kun arvot eivät korreloi keskenään kovin paljon.)

ALKUARVOT
beeta   t  -->      W    p-value
0.0    0.0     0.9035  6.009e-12
0.0    1.0     0.9417  1.032e-08
4.0    0.0     0.9437  1.662e-08
4.0    1.0     0.8806  1.715e-13

(JOHTOPÄÄTÖS: muuttujan beeta_39 reunajakauma ei ole normaalijakautunut.)
```

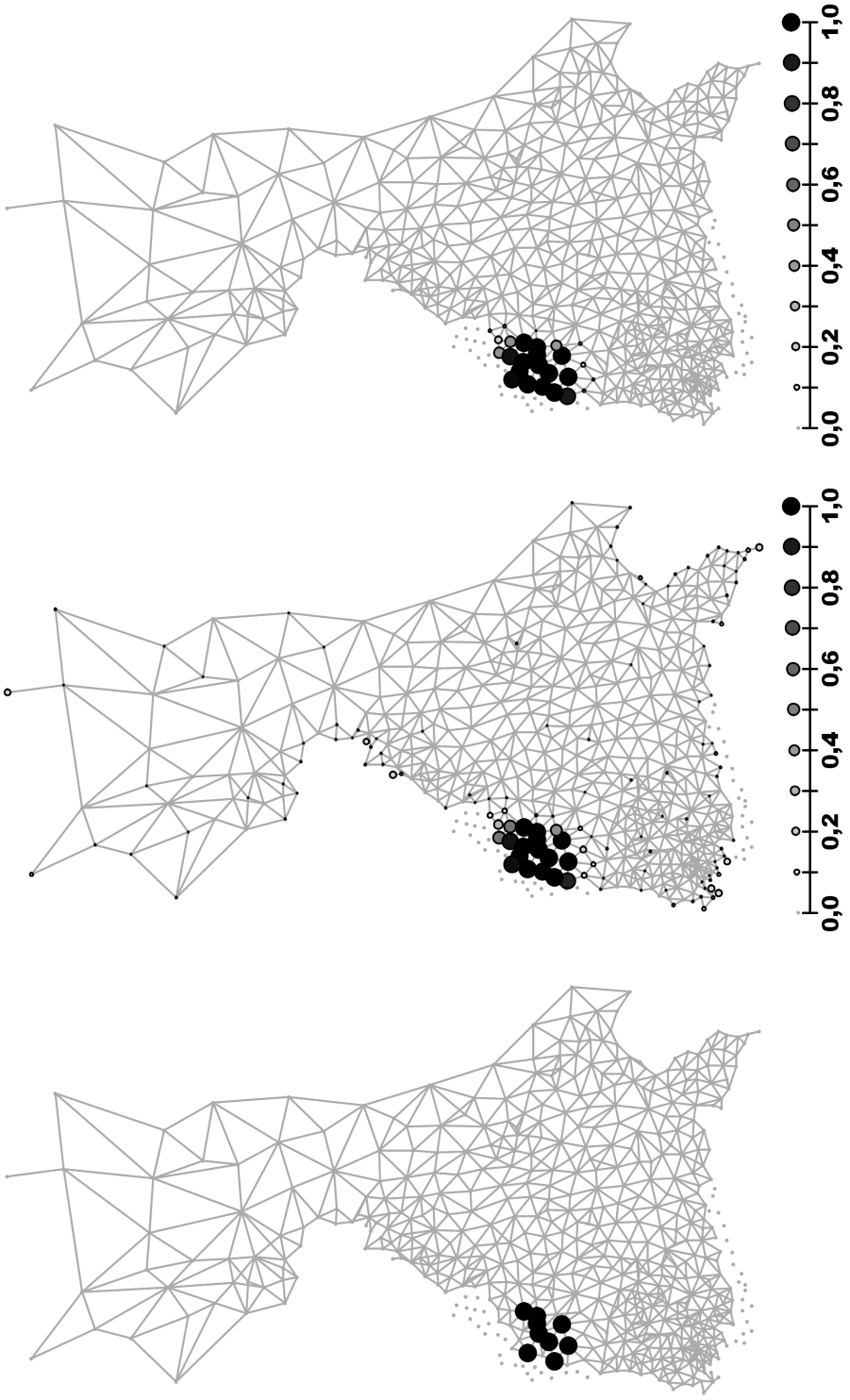
Liite 3. Murresana-aineisto ja sanojen käytön todennäköisyydet



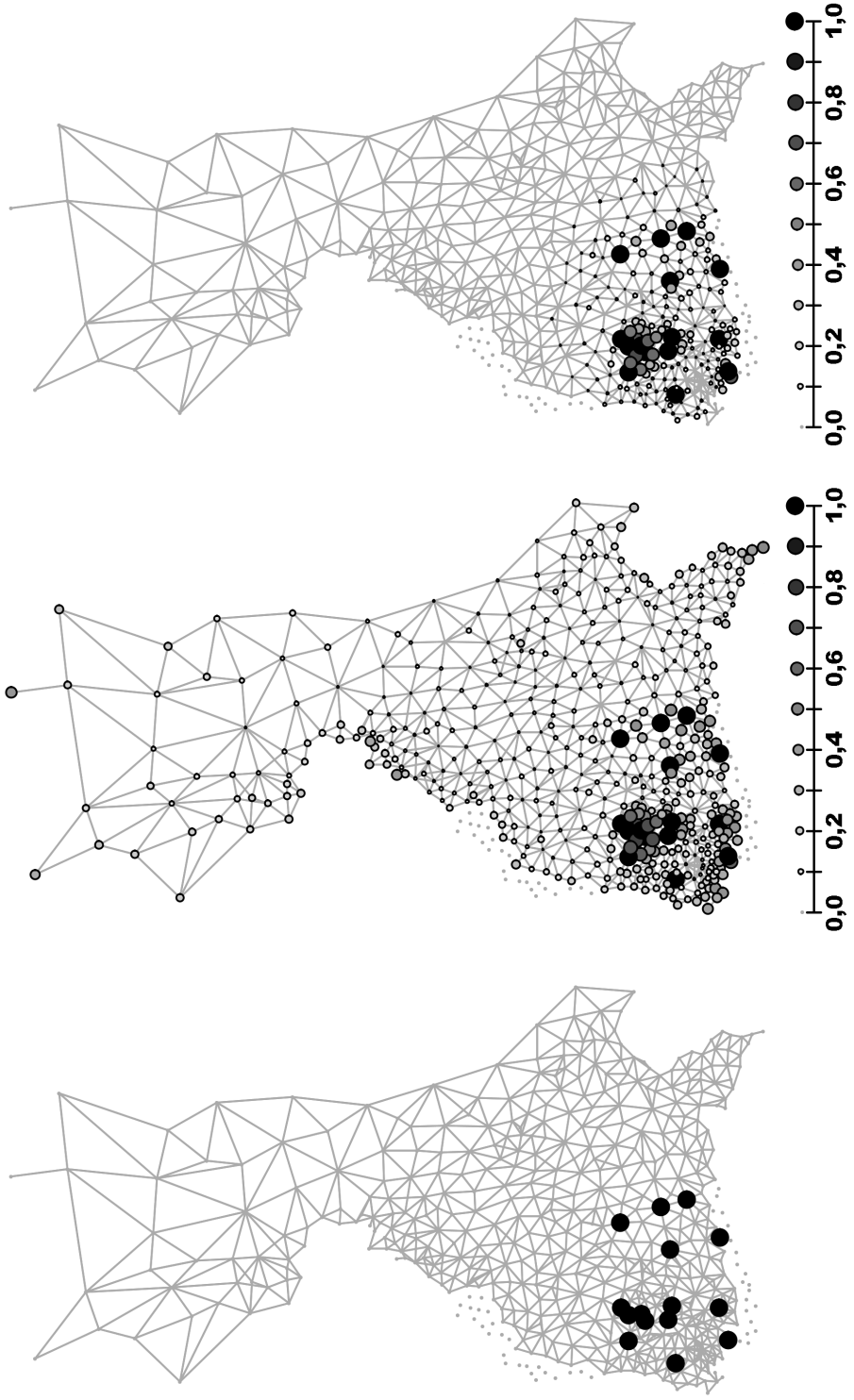
Kuva 1: Kuntien naapuriverkko, jossa solmut ovat kuntien keskipisteitä. Nyky-Suomen raja on suuntaa antava ja testiin kuulumattomat kunnat on jätetty naapureitta.



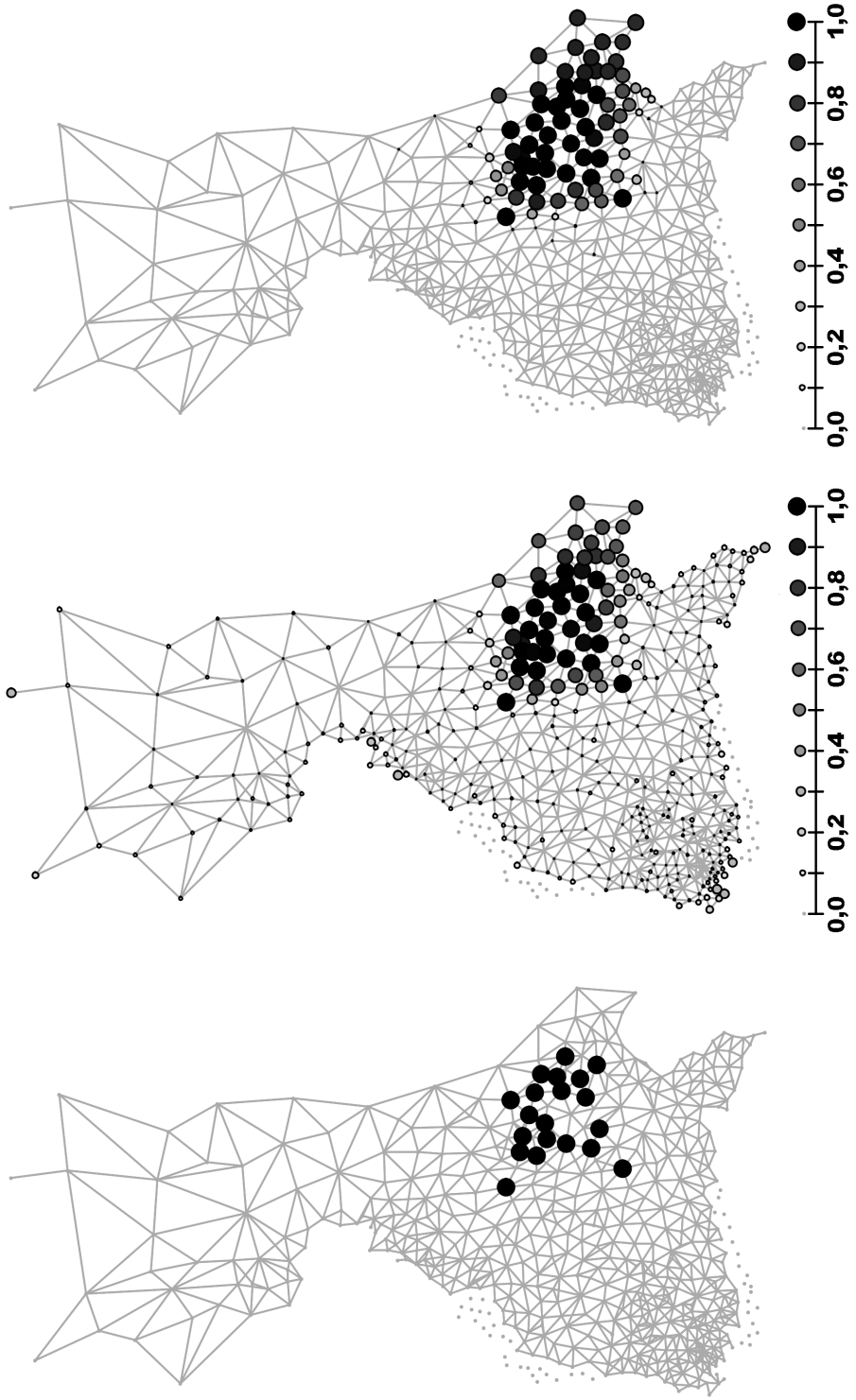
Kuva 2: Kunnista kerättyjen murresanojen lukumäärät.



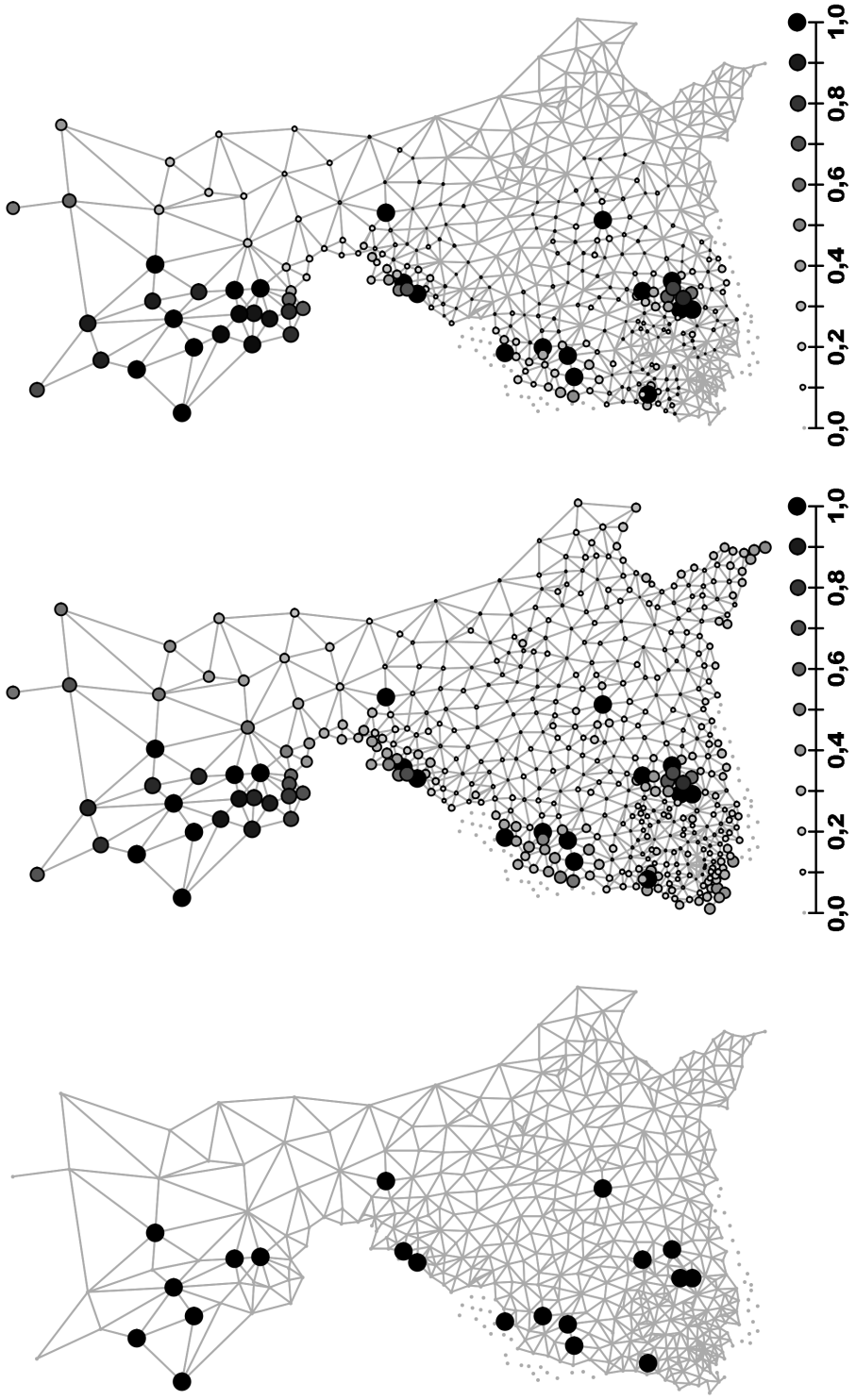
Kuva 3: Murresanan 136:*alaa*- havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{136}] = 2{,}63$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{136}] = 2{,}11$. Alhaalla alkuperäiset havainnot (9 kpl). (Sanan 136:*alaa*- kuvaus: *ala*.)



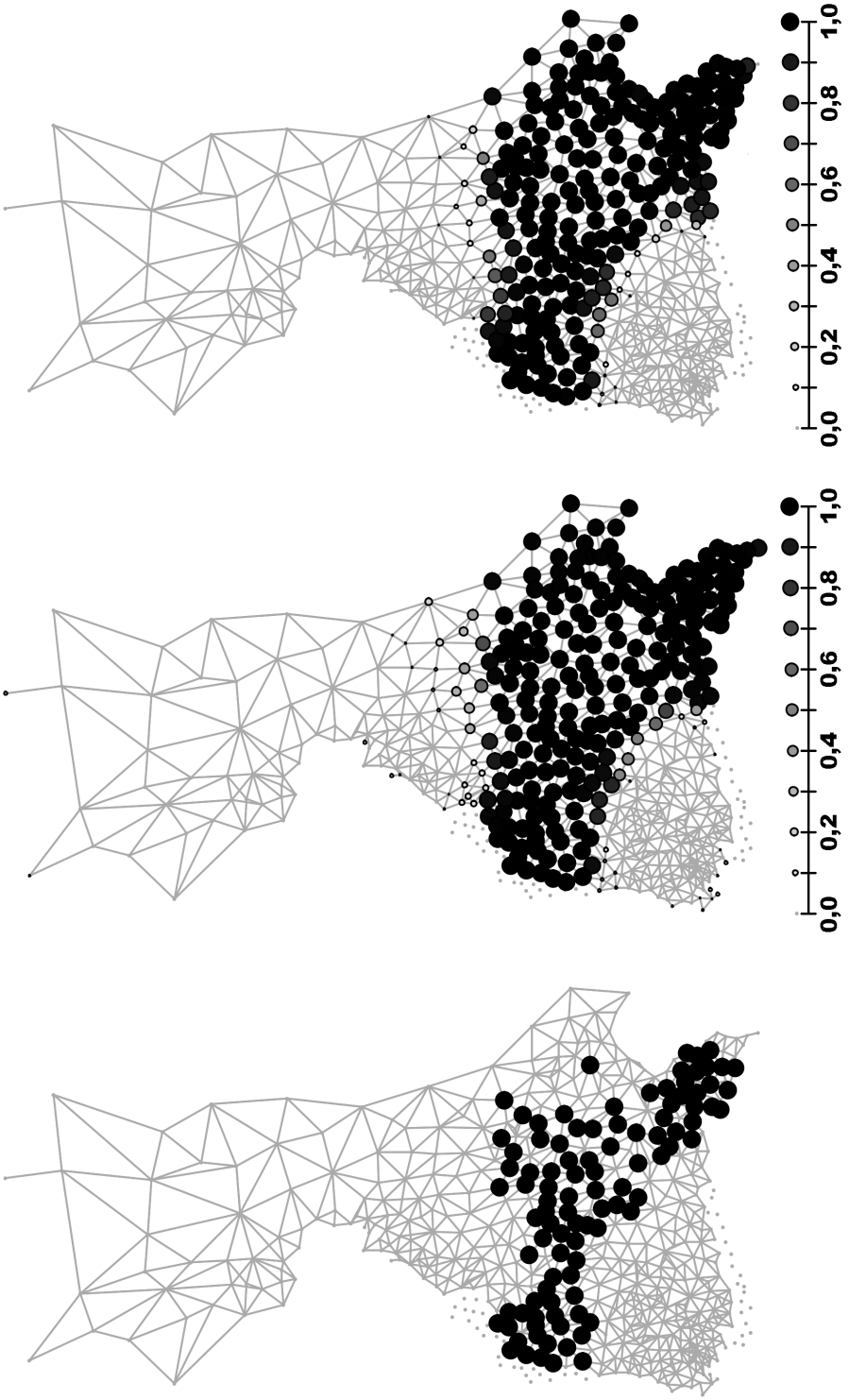
Kuva 4: Murresanan 156:*alasmaa* havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{156}] = 0{,}75$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{156}] = 0{,}58$. Alhaalla alkuperäiset havainnot (15 kpl). (Sanan 156:*alasmaa* kuvaus: *alamäki*, käyttö: "*Peestin sen (hevosen) tota alasmaat meneme.*")



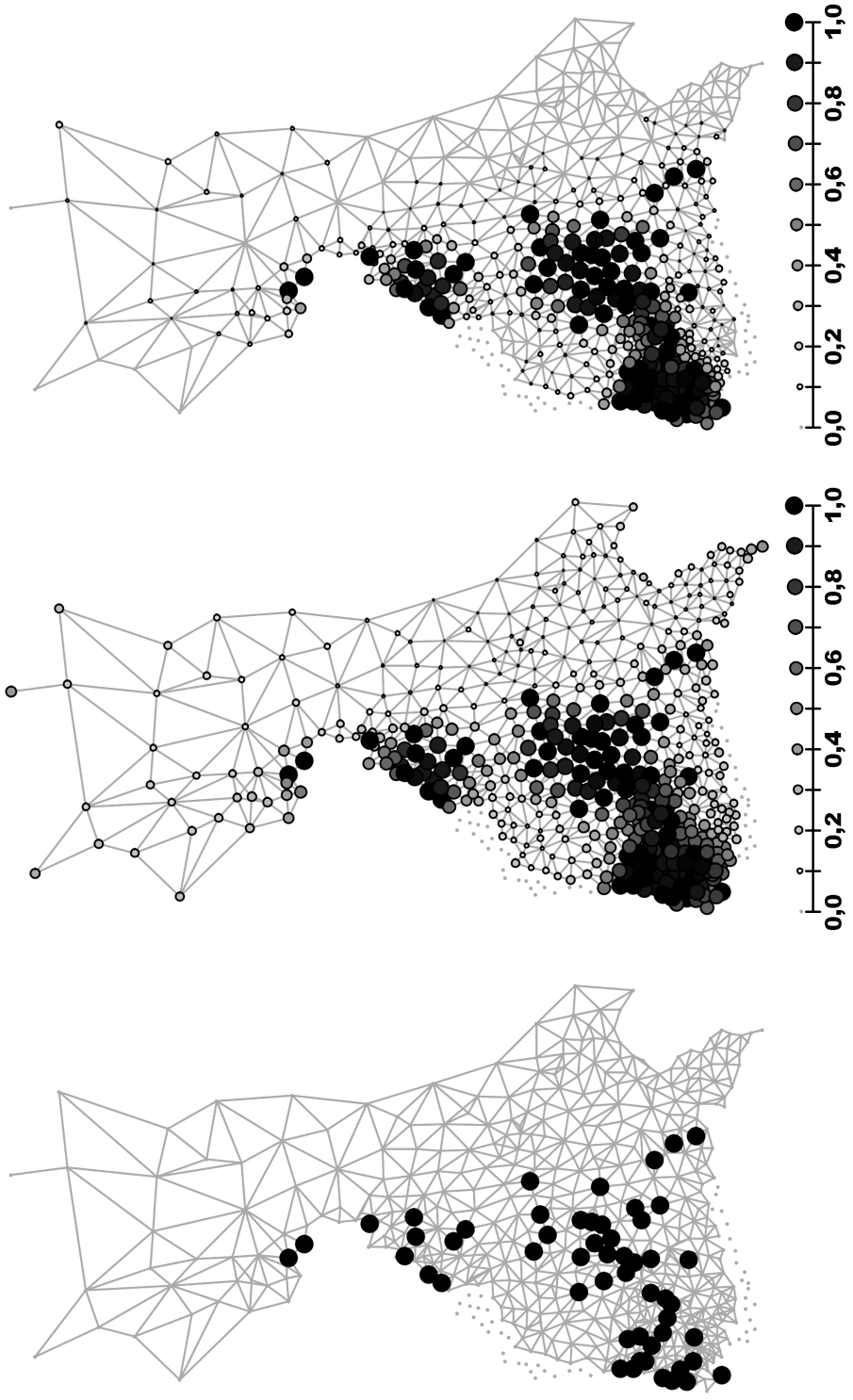
Kuva 5: Murresanan 113:*ajelus* havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{113}] = 1{,}74$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{113}] = 1{,}10$. Alhaalla alkuperäiset havainnot (20 kpl). (Sanan 113:*ajelus* kuvaus: *kulku talosta toiseen keräämässä ruoan, rehun ja vaatteen apua tms.; siten saatu anti*, käyttö: "*Köyhät vaimot kävivät laskiaist ajelemas, keräämässä taloista laskiaisena herneitä, sianlihaa ym.*")



Kuva 6: Murresanan 133:*akorti* havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{133}] = 0{,}70$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{133}] = 0{,}56$. Alhaalla alkuperäiset havainnot (20 kpl). (Sanan 133:*akorti* kuvaus: *(urakka-)sopimus*, käyttö: "*No net on kirjalisen akortin tehnee vissiin.*", "*Mää tein akortin, et otas ser rakennuksen tehräksen.*")



Kuva 7: Murresanan 135:*ala* havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{135}] = 2{,}60$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{135}] = 4{,}76$. Alhaalla alkuperäiset havainnot (99 kpl). (Sanan 135:*ala* kuvaus: *alle*, käyttö: "*Paa se hiirenpyytö latinkii tuonne kuapi ala.*")



Kuva 8: Murresanan 260:*aluton* havainnot ja käytön todennäköisyydet. Ylhäällä todennäköisyydet kattavuuden kolme paikallisella simulaatiolla, jossa $\mathtt{E}[\beta_{260}] = 0{,}73$. Keskellä todennäköisyydet tavallisella simulaatiolla, jossa $\mathtt{E}[\beta_{260}] = 0{,}72$. Alhaalla alkuperäiset havainnot (54 kpl). (Sanan 260:*aluton* kuvaus: *perätön, valheellinen*, käyttö: "*Se on vallan alutont, ei siin ol mittääm perrää.*")