TEKNILLINEN KORKEAKOULU
Sähkö- ja tietoliikennetekniikan osasto

Antti Ajanki

Geenisäätelyn mallinnus tilanneriippuvilla Bayes-verkoilla

Diplomityö, joka on jätetty opinnäytteenä tarkastettavaksi diplomi-insinöörin tutkintoa varten Espoossa 30.10.2006

Työn valvoja: Professori Samuel Kaski
Työn ohjaaja: TkT Janne Nikkilä

Tekijä:	Antti Ajanki	
Työn nimi:	Geenisäätelyn mallinnus tilanneriippuvilla Bayes-verkoilla	
Päivämäärä:	30.10.2006	**Sivuja:** 87
Osasto:	Sähkö- ja tietoliikennetekniikka	
Professuuri:	T-61 Informaatiotekniikka	
Työn valvoja:	Prof. Samuel Kaski	
Työn ohjaaja:	TkT Janne Nikkilä	

Solut tulevat toimeen useissa erilaisissa olosuhteissa, koska niiden toimintaa ohjaavien geenien ilmentymisaktiivisuus voi muuttua ympäristöstä tulevien signaalien tai toisten geenien tuottamien proteiinien vaikutuksen perusteella. Geenien väliset säätelysuhteet määräävät solun käyttäytymisen. Säätelyverkoston koko ja monimutkaisuus tekevät sen selvittämisestä haastavan ongelman.

Todennäköisyyslaskentaan perustuvat Bayes-verkot ovat eräs yleisesti käytetty esitystapa geenien säätelysuhteiden matemaattiseen mallintamiseen. Niille on olemassa opetusalgoritmeja, jotka etsivät mitattuihin ilmentymisprofiileihin parhaiten sopivan verkon. Opitun verkon rakenne voidaan tulkita geenien säätelyverkoksi.

Yleensä Bayes-verkkojen opetusmenetelmät olettavat, että kaikki havainnot on tehty samoissa olosuhteissa. Jos halutaan tutkia miten säätelyvuorovaikutukset muuttuvat olosuhteiden välillä, eräs tapa olisi opettaa erilliset verkot kuvaamaan eri olosuhteiden havaintoja ja verrata opittuja verkkoja keskenään. Silloin kunkin verkon opetukseen olisi kuitenkin käytettävissä vain osa opetusnäytteistä, mikä saattaisi johtaa ylisovittumiseen.

Tämä työ esittelee verkkorakenteen ja opetusalgoritmin, joita voidaan käyttää säätelyerojen etsimiseen. Näytteen mittausolosuhde huomioidaan itsenäisenä luokkamuuttujana. Uutta työssä on tapa, jolla luokkaa käytetään määräämään solmujen jakaumien riippuvuudet. Se helpottaa opitun verkon tulkintaa. Luokkamuuttujan ansiosta kaikki riippuvuudet voidaan esittää yhdessä verkossa, jonka opetukseen voidaan käyttää kaikkia havaintoja. Esiteltävä opetusalgoritmi löytää automaattisesti ne verkon osat, joissa on eroja luokkien välillä.

Työssä osoitetaan keinotekoisia opetusnäytteitä käyttäen, että ehdotettu opetusalgoritmi tuottaa paremmin oikeaa vastaavia verkkoja kuin oman verkon opettaminen erikseen joka olosuhteelle. Menetelmää sovelletaan stressaavien olosuhteiden aiheuttamien säätelyerojen etsimiseen hiivassa.

Avainsanat: Bayes-verkko, rakennehaku, geenien ilmentyminen

Author:	Antti Ajanki	
Name of the Thesis:	Modeling of gene regulation with context dependent Bayesian networks	
Date:	Oct 30, 2006	**Number of pages:** 87
Department:	Electrical and Communications Engineering	
Professorship:	T-61 Computer and Information Science	
Supervisor:	Prof. Samuel Kaski	
Instructor:	Janne Nikkilä, D.Sc. (Tech.)	

Cells of an organism survive in many kinds of environments because the expression of the genes governing all activities in the cells are affected by signals from external environment and proteins produced by other genes. This means that the gene regulating relationships form a complex network, which controls the behavior of the cell. The size and complexity of the network make deciphering it a challenging problem.

Bayesian networks which are based on probabilistic modeling are a commonly utilized framework for modeling gene regulating relationships. Structure learning algorithms are used to discover the network which best fits to the measured expression profiles. The learned network structure can be interpreted as a map of gene regulation.

Usually the learning algorithms for the Bayesian networks assume that all observations have been measured under one condition. Learning distinct networks for each environmental condition would provide a way to study how regulation differs between the conditions. However, this would mean that only a subset of the observations would be available for training each of these networks.

This work presents a network structure and a learning algorithm which are intended for finding differences in the regulation between different conditions. The condition of the observation is handled as a class variable. The novel way of determining the dependencies of the variables based on the class makes the interpretation easier. Because the class of the observation is just a node in the network all dependencies can be represented as one network which can be learned using all the samples. The proposed algorithm automatically discovers the class dependent sections of the network.

Using synthetic training examples, it is shown that the networks discovered by the new algorithm are closer to the ground truth than networks trained for each class separately. The method is applied to discovering differences in regulation between stressful and normal conditions in yeast.

Keywords: Bayesian networks, structure learning, gene expression

Alkulause

Tämä diplomityö on tehty Teknillisen korkeakoulun ja Helsingin yliopiston yhteisessä Statistical Machine Learning and Bioinformatics -ryhmässä. Työ on rahoitettu Teknologian kehittämiskeskuksen NeoBio-teknologiaohjelman Symbolic-projektista ja professori Kasken Helsingin yliopiston apurahasta.

Haluan kiittää työn valvojaa professori Samuel Kaskea ja ohjaajaa TkT Janne Nikkilää lukuisista neuvoista ja kommenteista työn eri vaiheissa. Erityisesti haluan kiittää perhettäni tuesta koko opintojen ajalta.

Espoossa, 30.10.2006

Antti Ajanki

Käytetyt merkinnät

α_{ijk}	Bayes-verkon Dirichlet-hyperparametri, joka liittyy solmun X_i arvoon k, kun isäsolmut ovat konfiguraatiossa j
$\boldsymbol{\theta}$	Paikallisten todennäköisyysjakaumien parametrien muodostama vektori
$\mathcal{A}$	Sijoittelufunktio, $\mathcal{A}(i) = m$, jos solmu X_i kuuluu moduuliin m
A_m	Moduuliin m kuuluvien solmujen joukko, $\{X_i \mid \mathcal{A}(i) = m\}$
D	Havaintovektorien joukko, $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(K)}\}$
N	Verkon solmujen (muuttujien) lukumäärä
N_{ijk}	Bayes-verkon paikallisen multinomijakauman tyhjentävä tunnusluku, eli niiden havaintojen lukumäärä, joissa muuttuja X_i on havaittu tilassa k samaan aikaan kun isäsolmut ovat olleet konfiguraatiossa j
$\mathcal{S}$	Bayes-verkon riippuvuusrakenne, eli lista $\{\boldsymbol{U}_1, \ldots, \boldsymbol{U}_N\}$ isäsolmuista jokaiselle verkon solmulle
$\boldsymbol{U}_i$	Solmun i isäsolmujen joukko
$\boldsymbol{U}_i'$	Moduulin i varsinaiset säätelijäsolmut, eli isäsolmut ilman luokkasolmua
$\boldsymbol{U}_i^c$	Moduulin i luokkaan c liittyvä luokkakohtainen isäsolmujen joukko
$X, Y, \ldots$	Satunnaismuuttujia
$\mathrm{Val}(X)$	Satunnaismuuttujan X arvojoukko

Sisältö

Luku 1

Johdanto

Useat eliöt ovat hyvin sopeutuvaisia ja pystyvät mukautumaan monenlaisiin olosuhteisiin. Esimerkiksi monet kasvit tulevat toimeen vaikka lämpötila tai saatavilla olevan veden määrä vaihteleekin suuresti. Sopeutuminen on mahdollista, koska solut pystyvät mukautumaan vallitseviin olosuhteisiin muuttamalla tuottamiensa proteiinien määriä. Proteiinit ohjaavat kaikkia solun toimintoja aineenvaihdunnasta vieraiden bakteerien neutraloimiseen.

Ohjeet erilaisten proteiinien tuottamiseen löytyvät geenien DNA-sekvensseistä. Geenin ilmentyminen on prosessi, jossa geenin DNA-sekvenssin kopioidaan lähetti-RNA:ksi, jonka perusteella valmistetaan uusi proteiini. Geenien ilmentymisaktiivisuutta säätelevät paitsi solun ulkopuolelta tulevat signaalit myös eräiden toisten geenien tuottamat proteiinit. Geenien väliset säätelysuhteet muodostavat monimutkaisen verkoston, josta tunnetaan tarkasti vain pieniä osasia. Jos säätelyverkko tunnettaisiin kokonaan, pystyttäisiin solun käyttäytyminen erilaisissa tilanteissa ennustamaan tarkasti.

Useat tutkijat ovat esitelleet tapoja päätellä solun säätelyvuorovaikutuksia mitattujen ilmentymisaktiivisuuksien perusteella. Eräs yleisesti käytetty matemaattinen malli säätelysysteemille on Bayes-verkko. Se perustuu hyvin tunnettuihin todennäköisyyslaskennan periaatteisiin ja sille on olemassa opetusmenetelmiä, jotka ottavat havaintoihin liittyvät epävarmuudet huomioon. Jos verkon solmut edustavat geenejä, niin opitun verkon kaaret voidaan tulkita geenien välisiksi säätelyvuorovaikutuksiksi.

Useimmissa tutkimuksissa kaikki mittaukset on suoritettu samoissa olosuhteissa ja tavoitteena on oppia tälle olosuhteelle ominainen säätelyverkko. Tämän työn kohde on hieman toinen. Erilaisissa ympäristöissä solun säätelyssä on pieniä eroja. Jotkin säätelyvuorovaikutukset kytkeytyvät pois päältä tai uusia käynnistyy. Todennäköisesti kuitenkin suurin osa säätelysuhteista pysyy muuttumattomina. Työn tarkoituksena on kehittää menetelmä, joka etsii tällaisia eroavaisuuksia säätelysuhteissa eri olosuhteissa suoritettujen ilmentymismittauksien perustella. Eräänä motivaationa säätelyerojen etsimiselle toimii mutatoidun solukannan vertaaminen alkuperäiseen. Useat tutkijat ovat

tehneet soluihin kohdistettuja mutaatioita, joiden tarkoituksena on ollut saada solu tuottamaan enemmän tiettyjä kemiallisia yhdisteitä kuten penisilliiniä tai mannoosia, mutta usein solut ovat pystyneet kompensoimaan mutaatioiden vaikutukset ja havaittavaa lisäystä ei ole syntynyt. Mutatoidun ja normaalin kannan säätelyerojen löytäminen kertoisi miten tämä kompensointi tapahtuu ja lisäisi tietoa säätelystä.

Aikaisemmat menetelmät eivät ole käyttäneet näytteen mittausolosuhdetta erillisenä luokkamuuttujana. Vaikka luokka olisikin periaatteessa helposti lisättävissä niihin, olisi olosuhderiippuvien säätelyvuorovaikutusten etsiminen lopputuloksesta silti hankalaa ilman tässä työssä esiteltävää tapaa käsitellä solmujen paikallisten jakaumien säännöllisyyksiä. Naiivi ratkaisu olisi opettaa oma verkko jokaiselle olosuhteelle ja verrata verkkoja keskenään. Oletettavasti tulokset olisivat kuitenkin epäluotettavia, koska opetusnäytteitä olisi käytössä vähemmän opetettavaa verkkoa kohden ja myös ne säätelysuhteet, jotka todellisuudessa eivät muutu tilanteiden välillä, pitäisi oppia erikseen kummassakin verkossa. Jos suurehko osa säätelysuhteista pysyy muuttumattomana, pystytään päällekkäisyyttä todennäköisesti käyttämään hyväksi mallin opetuksessa.

Tämän työn tarkoituksena on toteuttaa Bayes-verkon opetusmenetelmä, joka automaattisesti löytää ne kohdat säätelyverkosta, joissa on eroja olosuhteiden välillä, mutta samalla käyttää kaikkia havaintoja muuttumattomana pysyvien kohtien opetukseen. Opetusalgoritmin hyvyyttä voi mitata käyttämällä opetukseen tunnetusta Bayes-verkosta generoituja näytteitä ja katsomalla kuinka lähellä generoinnissa käytettyä verkkoa opittu verkko on. Uutta mallia verrataan malliin, jossa joka luokalle opetetaan oma verkko, tutkimalla molempien menetelmien oppimien verkkojen samankaltaisuutta generoivan verkon kanssa. Erityisesti pienillä näytemäärillä uuden mallin pitäisi olla parempi, koska se pystyy paremmin hyödyntämään kaikki opetusnäytteet.

Seuraavassa luvussa selvitetään työn biologista taustaa. Luku 3 käsittelee matemaattisia malleja, joita on käytetty geenisäätelyn kuvaamiseen, keskittyen erityisesti Bayes-verkkoihin. Neljännessä luvussa esitellään uusi, säätelyn erojen käsittelemiseen sopiva Bayes-verkkomalli ja opetusalgoritmi sille. Kaksi seuraava lukua kertovat suoritetuista kokeista keinotekoisesti muodostetuilla opetusnäytteillä ja todellisilla ilmentymismittauksilla. Lopuksi tulee yhteenveto ja johtopäätökset.

Luku 2

Geenien ilmentymisen säätely

Tässä luvussa perehdytään aluksi siihen, miten solu valmistaa proteiineja geeneihin koodatun informaation perustella ja miten geenien aktiivisuutta mitataan. Aliluku 2.4 esittelee geeniontologian, joka ryhmittelee geenejä niiden tehtävien samankaltaisuuden perusteella, ja lopuksi perehdytään lyhyesti siihen miten päätellään, mitkä geenit voivat säädellä muiden geenien ilmentymistä.

Tässä työssä käytetään malliorganismina leivontahiivaa *Saccharomyces cerevisiae*. Hiiva on aitotumallinen, eli sen soluissa DNA sijaitsee tumakalvon rajaamassa alueessa toisin kuin yksinkertaisimmilla bakteereilla ja arkkieliöillä, joilta puuttuu tuma ja muita kehittyneitä soluelimiä. Yksisoluisena hiivan kaikki solut ovat keskenään samankaltaisia ja hiiva lisääntyy jakautumalla. Monisoluisten eliöiden solut erikoistuvat kehittyessään erilaisiksi kudoksiksi. Hiivan säätelyvuorovaikutukset ovat huomattavasti monimutkaisempia kuin esitummallisilla eliöillä, mutta kuitenkin yksinkertaisempia kuin monisoluisilla eliöillä [31], minkä takia hiiva on suosittu organismi säätelyverkkojen oppimiseen tarkoitettujen menetelmien testaamiseen.

2.1 Geenien ilmentyminen

Eliöiden perimä on tallennettu deoksiribonukleiinihappo- eli DNA-makromolekyyleihin. Eliön jokaisessa solussa on kopio DNA:sta. DNA:n sisältö ohjaa solujen, ja siis koko eliön, kehitystä.

Solussa DNA on tyypillisesti kiertynyt vähän tilaa vievälle kaksoiskierteelle. Kumpikin DNA:n kahdesta juosteesta koostuu typpiemäksistä, joita on neljää tyyppiä; adeniini (lyhenteenä A), guaniini (G), sytosiini (C) ja tymiini (T). Nämä sitoutuvat toisen juosteen vastaavalla paikalla sijaitseviin emäksiin vetysidoksilla, mutta vain neljä eri sidosta on mahdollisia: A-T, T-A, G-C ja C-G.

DNA voidaan ajatella hyvin pitkäksi ”merkkijonoksi”, joka koostuu neljästä

emäsparien esittämästä kirjaimesta. Tähän merkkijonoon on kirjattu tieto solujen tarvitsemien proteiinien eli valkuaisaineiden rakenteesta. Proteiinit toimivat solun rakennusaineina, viestien välittäjinä, solun toimintojen säätelijöinä ja ylipäätään ohjaavat solun käyttäytymistä ja kasvua. DNA:n pätkää, joka sisältää proteiinin valmistusohjeet, kutsutaan geeniksi[1]. Vain osa DNA:sta koodaa geenejä. Loppuosan, jota voi olla suurin osa koko DNA:n pituudesta, tarkoitusta ei tunneta tai sen epäillään olevan jäänteitä aikaisemmista evoluution vaiheista.

Tapahtumaa, jossa solun koneisto lukee yhden geenin sisältämän geneettisen informaation ja tuottaa uuden proteiinin, kutsutaan proteiinisynteesiksi tai proteiinin ilmentymiseksi. Ilmentyminen käynnistyy, kun prosessin aloittava RNA-polymeraasi, joka on DNA:n kopiointiin pystyvä proteiiniyhdiste, sitoutuu biokemiallisesti DNA:han hieman ennen geenin alkua merkitsevää emäsketjua sijaitsevalle niin sanotulle promoottorialueelle. Kiinnittymisen jälkeen polymeraasi avaa kaksoiskierteen ja kopioi geenin emäsjonon lähetti-RNA:ksi, joka on DNA:ta huomattavasti lyhyempi ja osittain eri emäksistä koostuva nukleiinihappo, ja joka toimii solussa geneettisen informaation siirtäjänä. Tämän alustavan vaiheen nimi on transkriptio. Lähetti-RNA:han voidaan tehdä joitain muokkauksia, mutta lopulta se kuljetetaan ribosomiin, joka on solun proteiineja valmistava tehdas. Ribosomi purkaa lähetti-RNA:n ja lukee sen välittämän geneettisen informaation ja kokoaa sen perusteella proteiinin. Tätä seuraa vielä proteiinin laskostuminen, eli sen muotoileminen oikeaan kolmiulotteiseen muotoonsa, ja kuljetus tarvittavaan paikkaan solussa.

Tällaisenaan kaikki solut tuottaisivat jatkuvana virtana geeniensä koodaamia valkuaisaineita. Todellisuudessa solujen toiminta voi muuttua huomattavasti solun kasvun eri vaiheissa tai ulkoisten olosuhteiden vaikutuksesta. Esimerkiksi monisoluisissa eliöissä eri kudosten solut ovat erikoistuneet hyvin erilaisiin tehtäviin vaikka kaikkia ohjaa kopio samasta DNA:sta. Tällainen dynaaminen toiminta on mahdollista, koska valkuaisaineiden valmistusta säädellään proteiinisynteesin jokaisessa vaiheessa. Säätelymekanismit riippuvat soluun ympäristöstä tulevista signaaleista ja solun sisäisistä vuorovaikutuksista. Ne antavat solulle kyvyn mukautua olosuhteiden muutoksiin. Ilman säätelyä solu toimisi jatkuvasti samalla tavalla.

Tärkein säätelyvaihe on transkriptio. Yksinään RNA-polymeraasi on varsin tehoton transkription ylläpitäjä. Yleensä sitä auttaa joukko *säätelytekijöitä*, jotka ovat geenin promoottorialueelle tai suoraan RNA-polymeraasiin sitoutuvia proteiineja. Ne joko nopeuttavat tai estävät geenin kopioitumista lähetti-RNA:ksi ja siten vastaavan valkuaisaineen tuotantoa. Useimmat säätelytekijät ohjaavat vain tiettyä solun toimintoa ja pystyvät siksi sitoutumaan vain joidenkin geenien promoottorialueelle. Pieni osa on kuitenkin globaaleja säätelijöitä ja vaikuttaa suureen joukkoon geenejä [30, 50]. Esimerkiksi kolibakteerin *Escherichia colin* noin 300 säätelytekijästä 7 vaikuttaa suoraan 51 prosenttiin

[1]Historiallisesti sanalla *geeni* on ollut myös toinen, abstraktimpi merkitys. Jo ennen DNA:n löytämistä geeni tarkoitti sukupolvelta toiselle periytyvän ominaisuuden, kuten silmien värin, välittäjää.

geeneistä [35]. Globaalit säätelytekijät pitävät huolen solun perustoimintojen tarvitsemien proteiinien tuottamisesta, muut säätelijät ohjaavat solun erikoistuneempia prosesseja, joita tarvitaan vain joissain tilanteissa.

Säätelytekijöiden sitoutumisen lisäksi muita tapoja ilmentymisen säätelyyn ovat esimerkiksi vaihtoehtoiset tavat leikellä lähetti-RNA:ta tai valmiin proteiinien laskostuminen vaihtoehtoiseen muotoon, jolloin proteiinin ominaisuudet muuttuvat, koska sen kyky sitoutua muiden kemiallisten yhdisteiden kanssa riippuu sen fyysisestä rakenteesta.

Useimmat säätelytekijät ovat proteiineja, joita solu itse pystyy valmistamaan. Koska myös niiden proteiinisynteesiä säädellään joko toisilla säätelytekijöillä tai solun ulkopuolisista olosuhteista tietoa välittävillä signalointiproteiineilla, muodostuu säätelyvuorovaikutuksista monimutkaisia ketjuja. Kokonaista säätelysysteemiä voi kuvata abstraktilla tasolla verkkona, jossa geenit on yhdistetty toisiinsa, jos ensimmäisen geenin tuottama proteiini säätelee toisen ilmentymistä. Tällainen abstrakti esitys on usein riittävä, vaikka todellisuudessa säätelyvuorovaikutukset toimivat aina proteiinien välityksellä.

2.2 Olosuhderiippuva säätely

Solu on dynaaminen systeemi, joka pystyy vastaamaan ulkoisiin ja sisäisiin olosuhteiden muutoksiin muuttamalla säätelyään tilanteen mukaan. Vain osa solun geeneistä on toiminnassa kullakin hetkellä ja olosuhteiden muutos vaikuttaa geeneihin käynnistäen tai sammuttaen osan säätelyvuorovaikutuksista.

Fry ja Farnham [17] listaavat erilaisia mekanismeja, jotka voivat muuttaa säätelyvuorovaikutuksia. Tällaisia ovat esimerkiksi erilaiset mutaatiot, jotka estävät säätelytekijän sitoutumisen, tai sairaudet, joissa tärkeän valkuaisaineen konsentraatio on muuttunut normaalitilanteeseen verrattuna. Joidenkin proteiinien täytyy toimia yhdessä jotta ne pystyisivät käynnistämään transkription, mutta yksinään ne eivät toimi tai vaikutus on hyvin heikko. Tällainen säätelyvuorovaikutus toimii vain, jos kaikki tarvittavat proteiinit ovat yhtäaikaa läsnä solussa. Joidenkin geenien promoottorialueelle voi sitoutua useita säätelytekijöitä [30]. Geenien ilmentymisaktiivisuus siis riippuu siitä mitä säätelytekijöitä solussa kulloinkin on.

Solun säätelyverkosto on hyvin monimutkainen systeemi. Osa monimutkaisuudesta on seurausta säätelyn päällekkäisyyksistä, joiden ansiosta solu on epäherkkä useiden mutaatioiden vaikutuksille. Bailey [3] huomauttaakin, että ajatus siitä, että yksi geeni olisi suoraan vastuussa jostain makroskooppisesta biologisesta ominaisuudesta, on liian yksinkertainen ja mikä tahansa yhteen geeniin kohdistettu mutaatio aiheuttaa yleensä huomattavia muutoksia useiden geenien ilmentymiseen. Todisteena tästä Bailey luettelee useita kokeita, joissa tutkijat ovat kohdistettujen mutaatioiden avulla nostaneet niiden hiivasolujen ilmentymistasoja, joiden on uskottu olevan keskeisiä jonkin kemiallisen yhdis-

teen, kuten penisilliinin, valmistusprosessissa. Yleensä kuitenkin kokeessa on havaittu, että mutaatiolla ei ollut toivottua vaikutusta ja mutatoitu kanta on käyttäytynyt lähes samalla tavalla kuin alkuperäinenkin, eli hiiva on jotenkin pystynyt kompensoimaan mutaation vaikutukset.

Tällaisten olosuhteista riippuvien säätelyvuorovaikutusten löytäminen motivoi luvussa 4 esiteltävän menetelmän, joka pystyy etsimään eroja säätelysuhteissa eri tilanteissa mitattujen ilmentymisprofiilien väliltä. Sitä voi käyttää esimerkiksi selvittämään miten säätelyvuorovaikutukset poikkeavat terveessä ja sairaassa solussa, mikä paljastaisi mahdollisia kohteita uusille lääkkeille. Toinen käyttötapa olisi selvittää millä tavalla solu kumosi tehdyn mutaation. Se antaisi biologeille uutta tietoa solujen käyttäytymisestä.

2.3 Ilmentymisen mittaaminen

Ensimmäisen aitotumallisen eliön genomi, eli DNA:n täydellinen emäsjärjestys, onnistuttiin kartoittamaan kokonaan vuonna 1996, kun useiden laboratorioiden yhteistyönä valmistui hiivan emäskartta. Tämän jälkeen on onnistuttu selvittämään myös useiden monisoluisten eliölajien genomeja. Pelkän DNA-sekvenssin tunteminen ei kuitenkaan ratkaise solun toiminnan mysteeriä, koska jo pelkästään geenien tunnistaminen DNA:sta on vaikeaa ja geenin koodaaman proteiinin rakenteen, ja siten myös sen sitoutumisominaisuuksien, ennustaminen on vielä hankalampaa.

Yleisin tapa tehdä päätelmiä valkuaisaineiden toiminnasta solussa on niiden konsentraation seuraaminen kokeellisesti erilaisissa olosuhteissa. Jos proteiinin määrässä esimerkiksi on huomattava ero eri happamuusasteisissa ympäristöissä suoritettujen mittausten välillä, niin kyseinen proteiini liittyy luultavasti solun happotasapainon ylläpitoon. Aikaisemmin mittaaminen on ollut hyvin hidasta, mutta DNA-sirutekniikka [33, 41, 19] on mahdollistanut jopa kymmenientuhansien erityyppisten lähetti-RNA-molekyylien pitoisuuden selvittämisen yhdellä mittauksella. Esimerkiksi kaikkien hiivan geenien lähetti-RNA-aktiivisuuden mittaaminen yhdellä kertaa tuli mahdolliseksi vuonna 1997 noin vuosi hiivan genomien kartoittamisen jälkeen. DNA-sirutekniikka onkin nopeuttanut huomattavasti useita biologisia mittauksia.

DNA-siru (kuva 1) on parin neliösenttimetrin laajuinen lasilevy, jolla on säännöllisessä hilassa jokaista mitattavaa lähetti-RNA:ta varten oma tuhansista identtisistä koettimista koostuva piste. Koetin on lyhyt (noin 20-200 emäsparia) komplementaarisen RNA:n tai DNA:n pätkä, johon tietty lähetti-RNA sitoutuu, jos sen emäkset sopivat yhteen koettimen emästen kanssa. Sirua valmistettaessa täytyy siis päättää mitä sillä halutaan mitata ja valita sopivia pätkiä koettimiksi. Aina sopivien koettimien valinta ei ole helppoa ja joskus käy ilmi, että koetin ei sidokaan sitä lähetti-RNA:ta jota oli tarkoitus. Usein käytetään valmiita kaupallisia DNA-siruja, mutta jotkin laboratoriot valmistavat omat sirunsa.

Kuva 1: Kaksi DNA-sirua ja mittakaavana tulitikku.[2]

Niin sanottuja kaksikanavasiruja käytettäessä mittausprosessin aluksi referenssikannasta ja käsiteltävästä kannasta erotellaan RNA-molekyylit. RNA-näytteet värjätään kemiallisesti, referenssi- ja koenäytteet eri väreillä. Näytteet yhdistetään ja huuhdotaan sirun yli, jolloin lähetti-RNA:t sitoutuvat koettimiin. Lopuksi siru asetetaan skanneriin, joka mittaa väriaineiden intensiteetit jokaisen koetinpisteen kohdalla. Lopputuloksena saadaan lukuarvoja, jotka kertovat jokaisen koettimen sitoman RNA:n pitoisuuden suhteessa referenssikannan pitoisuuteen. Koska koettimia voi olla tuhansille geeneille, DNA-sirumittauksista saatavan valtavan tietomäärän käsitteleminen vaatii tarkkaa tilastollista analysointia, muuten oleelliset tulokset voivat hukkua tietomassaan.

DNA-sirumittaustuloksissa on paljon kohinaa johtuen mm. käytettävän solunäytteen epähomogeenisuudesta, itse sirun epäpuhtauksista ja solun sisäisen tilan vaihteluista mittausten välillä. Toistokokeilla ja tilastollisilla normalisoinneilla voidaan kuitenkin luotettavasti havaita noin 1,2-kertaisia muutoksia lähetti-RNA-pitoisuuksissa kahden näytteen välillä [19].

Kuten aikaisemmin mainittiin, joidenkin geenien ilmentymistä säädellään vielä senkin jälkeen, kun ribosomi on ottanut vastaan lähetti-RNA:n. Tämän vuoksi lähetti-RNA-pitoisuudet, joita DNA-sirutekniikka mittaa, eivät suoraan vastaa proteiinipitoisuuksia. Kuitenkin nimenomaan proteiinien määrä on mielenkiintoinen, koska proteiinit vuorovaikuttavat fyysisesti solun osien kanssa. Todellisten proteiinikonsentraatioiden mittaamiseen on olemassa tekniikoita, mutta tällä hetkellä ne ovat huomattavasti kalliimpia ja epätarkempia kuin DNA-sirut eikä niitä siksi käytetä yhtä paljon. DNA-sirumittauksia voidaan kuitenkin pitää kohtuullisen luotettavina indikaattoreina proteiinipitoisuuksil-

[2]Kuvan on luovuttanut Creative Commons Attribution 2.5 lisenssin mukaiseen vapaaseen käyttöön Wikimedia Commonsin käyttäjä Schutz.

le, koska tiedetään, että suurin osa ilmentymisen säätelystä tapahtuu transkriptiossa [51], kun geenin informaatio kopioidaan lähetti-RNA:ksi.

DNA-siruja käytetään esimerkiksi etsittäessä geenejä, joiden toimintaan sairaus vaikuttaa. Geenit, jotka käyttäytyvät selvästi poikkeavalla tavalla terveestä ja sairaasta kudoksesta otetuissa näytteissä, ovat mielenkiintoisia lääkkeiden kehittämisessä, koska niihin kohdistetut lääkkeet todennäköisesti vaikuttavat tehokkaasti sairauden parantamiseen.

2.4 Geenien toiminnalliset ryhmät

Harvat proteiinit toimivat yksinään. Yleensä tietyn tehtävän suorittaminen vaatii monen proteiinin koordinoitua yhteistoimintaa. Samankaltaisissa tehtävissä toimivia proteiineja ja niitä tuottavia geenejä tarkastellaankin siksi usein ryhminä.

Geeniontologia [10] pyrkii kokoamaan yhteen tietokantaan biologista tuloksia geenien ominaisuuksista. Geeniontologia koostuu luettelosta termejä. Jokaiseen termiin tutkijat ovat liittäneet joukon geenejä. Esimerkiksi luokkaan *kylmyysvaste* kuuluu geenejä, jotka aktivoituvat, kun eliö joutuu normaalia kylmempään ympäristöön. Käytetty termistö on tarkasti kontrolloitua ja se on pyritty tekemään yksikäsitteiseksi, jotta esimerkiksi etsittäessä tietoa solujen kuolemasta osa ei jää löytymättä siksi että tiedon syöttäjä on käyttänyt sanaa *apoptoosi* eikä tarkempaa termiä *tyypin yksi ohjelmoitu solukuolema*.

Termit jakautuvat kolmeen kategoriaan: solun osat, biologiset prosessit ja molekyylien toiminnot. Sama geeni voi esiintyä useissa kategoriassa, jos esimerkiksi tiedetään, että geenin tuottama proteiini osallistuu hapen kuljettamiseen solun sisälle (molekyylitoiminto) ja proteiini toimii tuman ulkopuolella solulimassa (solun osa). Termit koostuvat tarkentavista termeistä, joten geeniontologia muodostaa puumaisen tietorakenteen. Esimerkiksi, edellä mainittu *kylmyysvaste* on osajoukko luokasta *lämpötilavasteet*, joka puolestaan kuuluu vielä yleisempään luokkaan *vasteet epäbiologisiin herätteisiin*.

2.5 Potentiaaliset säätelytekijät

Vain pieni osa valkuaisaineista säätelee geenien ilmentymistä. Suurin osa toimii solun rakennusaineena, viestien kuljettajina tai muissa tehtävissä. Lisäksi osa säätelijöistä voi toimia vain tietyssä tilanteessa tai vain osana suurempaa proteiinikompleksia.

Niiden geenien, joiden tuottamien proteiinien on ylipäätään mahdollista jossain tilanteessa toimia säätelijänä, tunteminen auttaa, kun etsitään uusia säätelyvuorovaikutuksia. Tiedon avulla pystytään yleensä nopeuttamaan hakua ja

rajaamaan täysin epärealistisia vaihtoehtoja pois. Potentiaalisia säätelytekijöitä voidaan yrittää tunnistaa esimerkiksi mittaamalla DNA-sekvenssin tai proteiinin rakenteen samankaltaisuutta tunnettuihin säätelytekijöihin [50]. Nämä menetelmät voivat kuitenkin löytää vain osan todellisista säätelijöistä, koska säätelytekijät voivat olla rakenteeltaan hyvinkin vaihtelevia.

Täydellisen säätelytekijälistan laatiminen on vaikeaa, mutta kohtuullisen hyviä approksimaatioita on mahdollista tehdä. Esimerkiksi Segal et al. [43] listaavat joukon geenejä, joiden proteiini voi mahdollisesti toimia säätelijänä. Listalle on otettu mukaan geenit, joiden aiemman biologisen tutkimuksen perusteella päätelty rooli viittaa siihen, että geenin tuottama proteiini on potentiaalinen säätelytekijä, ja proteiinit, jotka voisivat rakenteensa perusteella sitoutua jonkin geenin promoottorialueelle. Jättämällä pois globaalit säätelytekijät, joita tarvitaan kaikkien proteiinien transkription käynnistämiseen, ja joita on solussa yleensä aina riittävästi eikä niillä siksi ole siis kovin suurta merkitystä säätelyyn, he löytävät 466 potentiaalista säätelytekijää hiivan reilun 6000 geenin joukosta.

Luku 3

Graafiset mallit säätelyn kuvaajina

Tässä luvussa käsitellään aluksi lyhyesti matemaattisten mallien muodostamista kokeellisten mittausten perusteella yleisesti minkä jälkeen keskitytään tarkemmin Bayes-verkkoihin ja erityisesti siihen miksi ne ovat hyviä malleja geenisäätelylle. Lopuksi esitellään kaksi hyvin toimivaa menetelmää, jotka käyttävät Bayes-verkkoja.

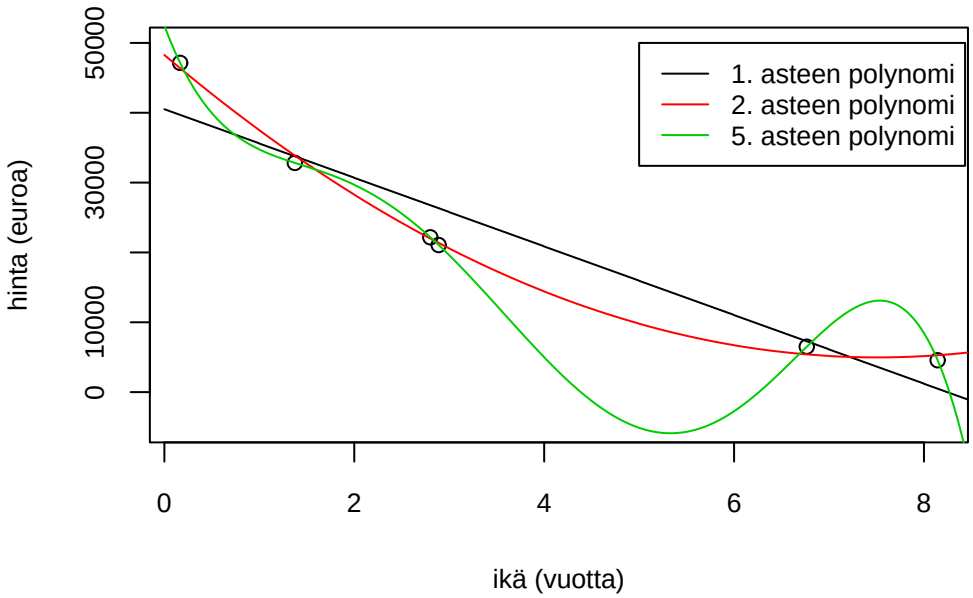
3.1 Mallin sovitus havaintoihin

Kokeellisissa tieteissä on tavoitteena ilmiön tai laitteen toiminnan ymmärtäminen mittauksia tekemällä ja yrittämällä päätellä niiden perusteella jotain systeemin käyttäytymisestä. Jotta havaintojen yleistäminen ennaltanäkemättömiin tilanteisiin olisi mahdollista, systeemi esitetään *matemaattisena mallina*, joka on yksinkertaistettu tai idealisoitu kuvaus systeemin toiminnasta. Sopiva abstraktiotaso sallii mallin käsittelyn matemaattisesti joko paperilla tai tietokoneella, mutta on samalla tarpeeksi tarkka tiivistäen oleellisen tiedon käsiteltävästä aiheesta ja mahdollistaen tulevien tilanteiden ennakoimisen.

Joissain tapauksissa mallin matemaattinen muoto on itsestään selvä ilmiötä kuvaavien fysiikan lakien tai muiden lainalaisuuksien perusteella. Esimerkiksi arvioitaessa putoamiskiihtyvyyttä pudottamalla pallo useita kertoja eri korkeuksilta ja mittaamalla putoamiseen kuluva aika on selvää, että pudotuskorkeus h ja aika t noudattavat (jos ilmanvastusta ja muita luultavasti vain vähän vaikuttavia tekijöitä ei huomioida) yhtälöä $h(t) = \frac{1}{2}at^2$, missä a on mallin *parametri*, fysikaaliselta merkitykseltään putoamiskiihtyvyys. Parametrin arvon estimoimiseksi mittauksista on olemassa useita menetelmiä, kuten suurimman uskottavuuden menetelmä, jossa valitaan se arvo, joka tekee havainnoista kaikkein todennäköisempiä käytetyn kohinamallin mielessä. Kun putoamiskiihtyvyys on estimoitu, voidaan mallin avulla tehdä ennustuksia (kuinka kauan kestää jos pallo pudotetaan kaksi kertaa korkeammalta?) tai päätelmiä mittausolosuhteista (putoamiskiihtyvyys Kuussa on pienempi kuin Maassa).

Aina mallinnettavalle ilmiölle ei ole yksinkertaista fysikaalista kuvausta, jota voitaisiin käyttää hyväksi. Tällöin täytyy valita havaintoja parhaiten selittävä malli kaikkien etukäteen mahdollisiksi katsottujen mallien joukosta eli *malliperheestä*. Mallin yhteensopivuutta havaintojen kanssa voidaan mitata erilaisilla tilastollisilla mittareilla. Joitain näistä käsitellään tarkemmin luvussa 3.2.2.

Eräs esimerkki mallinvalinnasta on havaintoihin sopivan käyrän valitseminen. Kuvassa 2 on kuvitteellisia käytettyjen autojen myyntihintoja iän funktiona. Jos halutaan arvioida miten auton arvo alenee ajan kuluessa voi myyntihintoihin yrittää sovittaa käyrän. Auton hinnan alenemiselle ei ole olemassa mitään fysiikan lakien tapaista universaalia lauseketta, siksi kuvassa on sovitettu eriasteisia polynomeja siten, että havaintojen ja käyrän välinen neliöllinen virhe minimoituu. Malliperheenä on siis polynomimuotoiset käyrät. Ensimmäisen asteen polynomi, jossa on vain kaksi parametriä, on liian jäykkä, se ohittaa useimmat havainnot melko kaukaa. Toisen asteen käyrä näyttää silmämääräisesti jo paljon paremmalta, mutta sekään ei kulje täsmälleen kaikkien havaintopisteiden kautta. Viidennen asteen polynomissa on tarpeeksi vapaita parametrejä, jotta se saadaan kulkemaan tarkalleen kaikkien havaintojen kautta, mutta havaintopisteiden välillä sen antamat ennusteet ovat selvästi järjettömiä. Sen esimerkiksi ennustaa noin viiden vuoden ikäisten autojen hinnan negatiiviseksi!



Kuva 2: Kuvitteellisia auton hintoja auton iän funktiona ja havaintoihin sovitettuja käyriä.

Vastaavanlaisiin ongelmiin törmätään aina sovitettaessa mallia, myös muita kuin yksiulotteisia käyriä, havaintoihin. Jos malli on tarpeksi monimutkainen suhteessa opetusnäytteiden määrään, on olemassa paljon parametrikombinaatioita, jotka sopivat yhtä hyvin näytteisiin. Koska opetusnäytteiden perusteella ei osata sanoa mitkä parametrien arvot ovat ”oikeita”, opitun mallin epä-

varmuus on suuri. Tästä johtuen mallin antamat ennusteet uusille, kaukana opetuspisteistä sijaitseville näytteille riippuvat siitä mitkä arvot parametreille valitaan ja ennusteet voivat olla pahastikin pielessä. Sanotaan, että malli on *ylisovittunut* ja että se ei pysty *yleistämään* opetusnäytteitä. Ylisovittumisen voi havaita esimerkiksi laskemalla virheen mallin ennustuksen ja oikean arvon välillä testinäytteille, joita ei ole käytetty millään tavalla hyväksi mallin opetuksessa. Ylisovittumista voi ehkäistä esimerkiksi tuomalla mukaan etukäteisoletuksia odotetusta käyttäytymisestä tai huomioimalla parametreihin liittyvän epävarmuuden jo sovitusvaiheessa.

3.2 Geenien ilmentymisen probabilistinen mallintaminen

DNA-mikrosirutekniikan kehittyminen on tehnyt mahdolliseksi solun proteiinien toiminnan tarkkailemisen laajassa mittakaavassa. Mitatuista ilmentymisprofiileista ei kuitenkaan näe suoraan mitkä säätelytekijät tai muut solun toiminnot ovat vaikuttaneet profiilin muotoutumiseen, koska yksittäisten säätelijöiden vaikutukset kasaantuvat ja hukkuvat valtavan tietomassan sekaan. Kehittämällä malli, joka kuvaa miten säätelijät vaikuttavat geeneihin ja toisiinsa, ja estimoimalla sen parametrit mittauksista voidaan säätelijöiden vuorovaikutuksista pystyä tekemään johtopäätöksiä.

Geenisäätelylle on esitelty lukuisia matemaattisia malleja [49]. Niistä suoraviivaisimmat ennustavat geenien ilmentymisaktiivisuutta regressiolla muiden geenien aktiivisuuden [46] tai muiden geenien aktiivisuuden ja promoottorialueen emäsjärjestyksen perusteella [40]. Niissä ajatuksena on muodostaa jokaiselle geenille oma regressori ja etsiä opetusvaiheessa joukko geenejä tai emäsjärjestyksiä, jotka ennustavat hyvin juuri sen aktiivisuutta. Tuloksena jokaiselle geenille saadaan lista säätelijöitä.

Toinen tapa on laskea havainnoista kaikkien geeniparien ilmentymisaktiivisuuksien korrelaatio [42] tai yhteisinformaatio [5]. Niiden parien, joille laskettu arvo on korkea, geenit ovat todennäköisesti suorassa vaikutussuhteessa keskenään ja parien, joille arvo on matala, geenit vaikuttavat toisiinsa korkeintaan epäsuorasti jonkin toisen geenin kautta. Vain suorat vuorovaikutukset ovat säätelyverkkoa rakennettaessa mielenkiintoisia, siksi lasketuista arvoista muodostetaan verkko yhdistämällä ne geenit, joiden parittainen korrelaatio tai yhteisinformaatio on tarpeeksi suuri. Raja-arvo valitaan siten, että valituksi tulevat vain ne parit, joille laskettu arvo poikkeaa merkitsevästi muiden parien arvoista sopivalla tilastollisella testillä mitattuna. Viitteen [5] menetelmä käyttää lisäksi epäsuorien yhteyksien karsimiseen tiedonkäsittelyepäyhtälöä, jonka mukaan kaikissa kolmen geenin ryhmissä pienin parittainen yhteisinformaatio liittyy todennäköisimmin epäsuoraan vuorovaikutukseen.

Kaikissa edellä esitellyissä menetelmissä tarkastellaan kerrallaan vain yhtä gee-

niä tai geeniparia. Niissä suorien ja epäsuorien vuorovaikutusten erottelu on vaikeaa. Seuraavaksi esiteltävissä malleissa koko verkko opitaan yhdellä kertaa, jolloin on ainakin periaatteessa mahdollista löytää myös oikeat epäsuorat vaikutukset. Yksityiskohtaisimmat kokonaisen verkon mallit perustuvat solun biokemiallisen toiminnan mallintamiseen hyvin tarkasti differentiaaliyhtälöillä [45, 53]. Ne eivät sovellu kovin hyvin ennaltatuntemattomien säätelysuhteiden etsimiseen, koska ne vaativat hyvin paljon ennakko-oletuksia ja toimivat kunnolla vain muutaman geenin verkoissa ennen kuin malleista tulee liian monimutkaisia. Toisessa ääripäässä ovat hyvin abstraktit mallit kuten Boolen verkot [1, 32], joissa geeni voi olla vain kahdessa tilassa, päällä tai pois päältä, ja geenien väliset vuorovaikutukset on rajoitettu loogisiksi Boolen funktioiksi. Näin abstrakti malli ei kovin helposti taivu selittämään tunnetusti monimutkaista geenisäätelyä. Eräs malli, jonka kompleksisuutta pystytään skaalaamaan melko vapaasti näiden ääripäiden välillä [20], on todennäköisyyslaskentaan perustuva Bayes-verkko. Siinä geenien ilmentymistasoilla voi olla enemmän kuin kaksi tilaa tai niitä voidaan mallintaa jatkuva-arvoisilla muuttujilla ja geenien väliset vuorovaikutukset voivat olla mielivaltaisia stokastisia funktioita.

DNA-sirumittaukset ovat tunnetusti varsin kohinaisia eivätkä ne välttämättä kerro suoraan proteiinien määrän säätelystä, koska sirut mittaavat lähetti-RNA:n pitoisuuksia mutta proteiinien tuotantonopeutta voidaan säädellä vielä RNA-vaiheen jälkeenkin. Todennäköisyyslaskentaan perustuvat mallit voivat jossain määrin ottaa nämä ongelmat huomioon. Niiden ajatuksena on mallintaa geenien ilmentymistasoja satunnaismuuttujina ja säätelyvuorovaikutuksia satunnaismuuttujien riippuvuuksina. Epävarmuutta kuvataan todennäköisyysjakaumalla; jos muuttujan arvo tunnetaan hyvin tarkasti, sen jakauma on kapea, ja vastaavasti, jos muuttujan arvoon liittyy suuri epävarmuus, jakauma on leveä. Todennäköisyysmallit pystyvät myös helposti käsittelemään puuttuvia mittauksia ja piilomuuttujia, mutta tässä työssä näitä ominaisuuksia ei käsitellä. Hyvin tunnettu todennäköisyyslaskennan teoria tarjoaa perustellut menetelmät mm. mallin valintaan, epävarmuuden käsittelyyn ja etukäteisoletuksien mukaan ottamiseen.

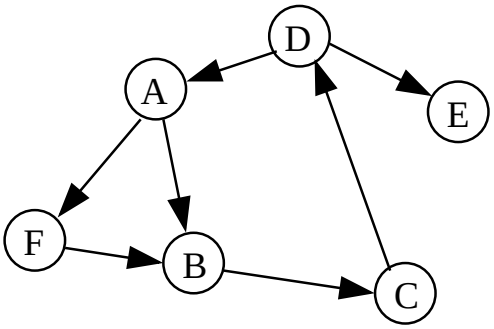
3.2.1 Todennäköisyysjakauma Bayes-verkkona

Satunnaismuuttujien $X_1, X_2, \ldots X_N$ käyttäytyminen ja niiden väliset riippuvuudet on täysin määrätty, jos tunnetaan niiden yhteistodennäköisyysjakauma $p(X_1, X_2, \ldots X_N)$. Se kertoo (diskreettiarvoisille muuttujille) jokaisen muuttujien kombinaation todennäköisyyden tai (jatkuva-arvoisille muuttujille) todennäköisyystiheyden. Satunnaismuuttujan X arvojoukkoa merkitään $\mathrm{Val}(X)$ ja sen saamaa arvoa pienellä kirjaimella $x \in \mathrm{Val}(X)$.

Yhteisjakauman esittämiseen tarvittavien parametrien määrä kasvaa hyvin nopeasti satunnaismuuttujien lisääntyessä. Esimerkiksi N:n diskreettiarvoisen muuttujan, joista kullakin on d mahdollista arvoa, yhteisjakauman esittämiseksi täytyy kertoa kaikkien d^N kombinaation todennäköisyydet. Jos nämä toden-

näköisyydet on tarkoitus estimoida äärellisestä määrästä havaintoja, on riskinä ylisovittuminen. Tällaisessa mallissa muuttujat voivat riippua mielivaltaisesti toisistaan. Todellisuudessa yleensä osa muuttujista on keskenään riippumattomia, jolloin yhteisjakaumassa osa parametreistä on samoja, eli *efektiivisten parametrien* määrä on pienempi. Bayes-verkko [37] on keino esittää tällaiset riippumattomuudet eksplisiittisesti helposti ymmärrettävässä muodossa.

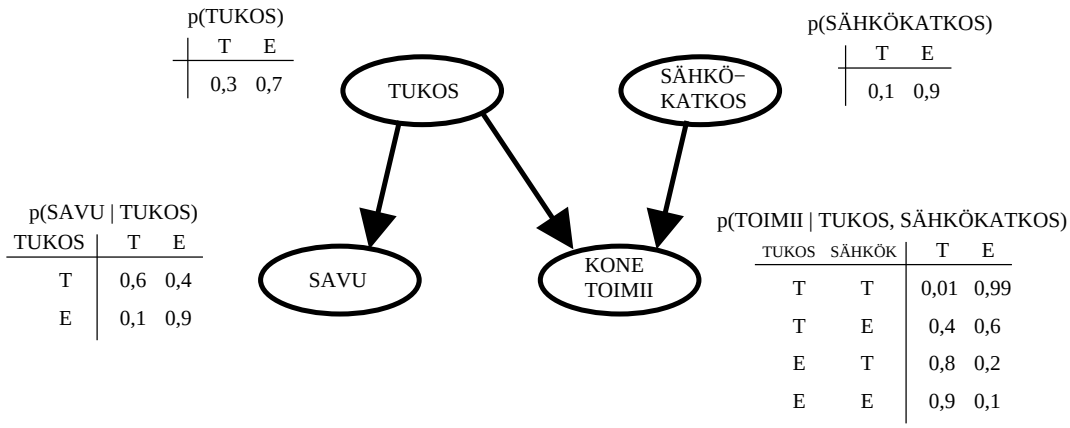
Bayes-verkko on suunnattu syklitön verkko, jonka solmut edustavat satunnaismuuttujia ja solmusta A on kaari solmuun B, jos muuttujan B arvot riippuvat suoraan muuttujan A arvoista. Tällöin sanotaan, että solmu A on B:n *isäsolmu* ja kääntäen B on A:n *lapsi*. Solmun X_i isäsolmujen joukkoa merkitään jatkossa $\boldsymbol{U}_i$:llä. Solmulla ei välttämättä tarvitse olla yhtään isäsolmua, eli $\boldsymbol{U}_i$ voi olla tyhjä joukko. Syklittömyys tarkoittaa, että lähdettäessä mistä tahansa verkon solmusta kulkemaan kaarien osoittamaan suuntaan ei ole mahdollista päätyä takaisin lähtösolmuun. Kuvassa 3 on esimerkki suunnatusta verkosta, jossa on yksi sykli (eli se ei ole Bayes-verkko).



Kuva 3: Suunnattu verkko, jossa solmut A, B, C ja D muodostavat syklin, mutta solmut A, F ja B eivät, koska kaarien suunnat eivät ole yhteensopivia. Syklisyyden takia tämä ei ole Bayes-verkko.

Verkon rakenne, eli verkkotopologia, kertoo vain muuttujien kvalitatiiviset riippuvuudet. Riippuvuuksien voimakkuudet esitetään jokaiseen verkon solmuun liittyvinä paikallisina ehdollisina todennäköisyysjakaumina $p(X_i|\boldsymbol{U}_i)$, jotka kertovat miten solmun muuttujan arvo riippuu isäsolmujen arvoista. Jakauma voi olla mikä tahansa todennäköisyysjakauma, mutta yleensä käytetään normaalijakaumaa jatkuva-arvoisille ja multinomijakaumaa diskreettiarvoisille muuttujille, koska silloin useimmat laskut pystytään suorittamaan suljetussa muodossa.

Kuvassa 4 on yksinkertainen esimerkki Bayes-verkosta, joka kuvaa paperikoneen vikaantumista. Koneen toimintaan vaikuttaa tässä tapauksessa kaksi tekijää. Jos paperikoneeseen tulee tukos, koneen toiminta voi estyä. Oikein paha tukos voi ilmetä koneesta nousevana savuna. Myös sähkökatkos voi sammuttaa koneen, jos tehtaan varageneraattorit eivät jostain syystä käynnisty. Mallissa jokaisella muuttujalla on kaksi mahdollista arvoa: tosi (T) ja epätosi (E). Jokaiseen solmuun liittyy multinomijakauma, jonka parametrit kertovat solmun muuttujan todennäköisyyden kaikilla isäsolmujen arvoilla.



Kuva 4: Esimerkki Bayes-verkosta.

Muodollisesti Bayes-verkko $\mathcal{B}$ on järjestetty pari $(\mathcal{S}, \boldsymbol{\theta})$, missä verkkorakenne $\mathcal{S}$ sisältää tiedon jokaisen solmun X_i isäsolmuista $\boldsymbol{U}_i$ ja parametrivektori $\boldsymbol{\theta}$ kertoo paikallisten todennäköisyysjakaumien muodon ja parametrit. Jos esimerkiksi paikalliset jakaumat ovat multinomijakaumia, niin vektori $\boldsymbol{\theta}$ sisältää todennäköisyydet $\theta_{ijk} = p(X_i = k|\boldsymbol{U}_i = j)$ kaikille muuttujien ja isäsolmujen arvojen yhdistelmille. Kun verkko ja paikallisten jakaumien parametrit tunnetaan, voidaan havaintovektorin $(x_1, x_2, \ldots, x_N)$ todennäköisyys laskea helposti tulona paikallisista termeistä:

$$p(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} p(x_i|\boldsymbol{u}_i).$$

Perinteisesti Bayes-verkkoja on käytetty päättelemään joidenkin muuttujien arvot, kun osa muista muuttujista on havaittu. Esimerkiksi kuvan 4 verkosta voitaisiin laskea todennäköisyys, että koneessa on tukos, jos siitä nousee savua, mutta kone kuitenkin toimii. Tavallisesti sovellusalan asiantuntija päättää Bayes-verkon rakenteen ja paikalliset todennäköisyydet ja käyttäjä kiinnittää havaittujen solmujen arvot mitattuihin arvoihin. Silloin muiden solmujen todennäköisyysjakauma voidaan päivittää verkon koodaamien riippuvuuksien mukaisesti. Verkon rakentaminen tällä tavoin on mahdollista vain, jos muuttujia on kohtuullisen pieni määrä, korkeintaan ehkä kymmenkunta. Suuremmilla verkoilla riippuvuuksien ja jakaumien päättäminen käy liian työlääksi ihmiselle. Bayes-verkkoja on sovellettu esimerkiksi sairauden diagnosoinnissa lääkäreiden apuna [4], avun etsimiseen tietokoneohjelmien ohjeista vapaalla teksihaulla [24] ja oppilaiden opitun ymmärtämisen testaamiseen [25].

Tässä työssä ei käsitellä edellä kuvatun kaltaista perinteistä päättelyä Bayes-verkossa vaan keskitytään vielä haastavampaan ongelmaan: verkon rakenteen oppimiseen havaintojen perusteella. Tämä on välttämätön tehtävä, jos muuttujia on liian paljon ihmisasiantuntijan hallittavaksi tai jos mielenkiinnon kohteena ovat muuttujien väliset riippuvuudet itsessään, eikä niinkään joidenkin muuttujien arvon ennustaminen havaittujen muuttujien perustella. Gee-

nisäätelyn mallinnus voidaan ymmärtää juuri tällaiseksi ongelmaksi. Tiedetään, että geenin ilmentymistasoon vaikuttaa muutama säätelijä. Toisaalta Bayes-verkossa jokaisen solmun arvo riippuu parista toisesta solmusta. Jos ilmentymismittausten perusteella opitaan Bayes-verkko, jossa solmut vastaavat geenejä, voidaan verkon riippuvuudet tulkita geenien välisiksi säätelyvuorovaikutuksiksi. Sopivasti opittavan verkon monimutkaisuutta rajoittamalla on mahdollista oppia jopa tuhansista solmuista koostuvia Bayes-verkkoja [38].

Bayes-verkot ovat määritelmänsä mukaisesti syklittömiä. Toisaalta tiedetään, että geenien säätelyverkoissa esiintyy takaisinkytkentöjä [30]. Siksi Bayes-verkkojen käyttäminen mallintamaan geenisäätelyä voi vaikuttaa ristiriitaiselta. Käytännössä näytteitä ei kuitenkaan ole koskaan tarpeeksi tarkan verkon oppimiseen, joten tuloksena on joka tapauksessa approksimaatio eikä syklien puuttuminen välttämättä ole kovin vakava ongelma. Puutteen vaikutuksia lieventää sekin tosiasia, että syklittömätkin Bayes-verkot pystyvät mallintamaan silmukoita sisältävän systeemin tasapainotiloja. Kaksi toisiaan syklisesti sääteleväää geeniä voivat esimerkiksi muodostaa kaksitilaisen kytkimen, jossa vain toinen geeneistä on aktiivinen kerrallaan [13]. Tällaista systeemiä on täysin mahdollista kuvata Bayes-verkolla.

Bayes-verkoista on myös kehitetty versio, jota voi käyttää syklien ja muiden dynaamisien ilmiöiden mallintamiseen, jos opetusnäytteet muodostavat tasavälisen aikasarjan. Dynaamisissa Bayes-verkoissa jokaisesta solmusta on kaksi kopiota, jotka vastaavat havaintoja muuttujan arvosta kahdessa peräkkäisessä aikapisteessä. Verkossa saa olla kaaria vain eri ajan hetkiin liittyvien solmujen välillä. Nämä kaaret esittävät muuttujien riippuvuuksia edeltävistä arvoista ja ne voivat esittää myös takaisinkytkentöjä vaikka itse verkon rakenteessa ei olekaan syklejä.

3.2.2 Verkon oppiminen mittausten perusteella

Bayes-verkon rakenteen oppimista kutsutaan todennäköisyyslaskennan teoriassa mallin valinnaksi. Tehtävänä on valita etukäteen kiinnitetystä malliperheestä parhaiten havaintoja kuvaava malli tai pieni joukko malleja. Yleisimmillään malliperhe voi koostua kaikista suunnatuista syklisistä verkoista, mutta joskus malliavaruutta voi rajoittaa etukäteistiedon perusteella. Esimerkiksi säätelyverkkoja opittaessa voi olla biologisesti perusteltua sallia vain verkot, joissa minkään solmun isäsolmujen lukumäärä ei saa olla annettua rajaa suurempi, koska todellisissa säätelyverkossa yhtä geeniä säätelee yleensä korkeintaan pari säätelytekijää. Malliperheen rajoittaminen tietenkin helpottaa mallinvalintaa, koska vaihtoehtoja on vähemmän, mutta liian tiukka rajoitus saattaa pudottaa hyvät vaihtoehdot pois.

Mallin valintaan sisältyy myös solmujen paikallisten jakaumien funktionaalisen muodon valinta. Käytännössä kaikille muuttujille valitaan parametriseltä muodoltaan sama jakauma. Yleensä käytetään multinomijakaumaa diskree-

teille satunnaismuuttujille tai normaalijakaumaa jatkuville muuttujille, koska niiden käsittely on helppoa. Jakaumien parametrit estimoidaan havainnoista.

Bayes-verkon mallinvalinta-algorimit voidaan jakaa kahteen pääluokkaan; riippuvuustesteihin ja optimointiin perustuviin menetelmiin. Riippuvuustesteihin nojaavissa menetelmissä lasketaan kaikkien muuttujaparien yhteisinformaatiot tai muut riippuvuusmitat, ja muodostetaan verkko asettamalla kaaret niiden parien välille, joiden riippuvuusmitta on tarpeeksi suuri. Lopuksi kaarille valitaan suunnat ja verkosta poistetaan syklit.

Tässä työssä keskitytään lähinnä optimointiin perustuviin menetelmiin, koska ne pystyvät ainakin teoriassa löytämään todelliset riippuvuudet toisin kuin riippuvuustestialgoritmit, jotka eivät osaa erotella epäsuoria riippuvuuksia suorista. Optimointialgoritmeissa keskeisessä asemassa on *yhteensopivuusmitta*, josta käytetään myös nimeä *pisteytysfunktio*. Se mittaa sitä kuinka hyvin annettu Bayes-verkko kuvaa havaintoja. Mittafunktion arvon tulisi olla suuri verkoille, joiden määrittämässä todennäköisyysjakaumassa tehtyjen havaintojen todennäköisyydet ovat korkeita, ja pieni verkoille, jotka saavat havainnot näyttämään epätodennäköisiltä. Tehtävänä on etsiä mitan maksimoiva verkko, jonka esittämä jakauma siis on mahdollisimman lähellä havaintojen empiiristä jakaumaa. Yleensä optimointi tapahtuu iteratiivisesti tekemällä verkkorakenteeseen pieniä muutoksia ja tarkistamalla kasvattavatko ne yhteensopivuusmittaa.

Parhaan verkon löytäminen on laskennallisesti hyvin vaativaa. Jos verkossa on N solmua, niin erilaisia verkkorakenteita on $N!$ kappaletta, mikä on liian paljon yksitellen läpikäytäväksi vähänkään isommalla N:n arvolla. Voidaan osoittaa [8], että jopa sellaisessa malliperheessä, jossa jokaisen solmun isäsolmujen määrä on rajoitettu kahteen, parhaan verkon löytäminen on NP-täydellinen ongelma, eli polynomiaikaista algoritmia ei luultavasti ole olemassa. Koska tarkan ratkaisun etsiminen on niin hidasta, melkein aina käytetään heuristisia optimointialgoritmeja, jotka löytävät yleensä hyvän ratkaisun suhteellisen nopeasti, mutta joiden tekemää virhettä ei voi helposti arvioida.

Yhteensopivuusmittoja

Yhteensopivuusmittoja on useita erilaisia. Minimikuvauspituuden periaatteen (MDL, minimum description length) [39] ajatuksena on mitata havaintojen ja mallin kuvaamiseen tarvittavaa informaation määrää ja valita se malli, jonka esittäminen yhdessä havaintojen kanssa on kaikkein taloudellisinta. Lam ja Bacchus [29] ovat esitelleet periaatteen mukaisen Bayes-verkon yhteensopivuusmitan. Jos verkko muokataan sellaiseksi, että se huomioi havainnoissa esiintyvät säännönmukaisuudet, niin havainnot voidaan esittää tiivisti. Toisaalta myös verkon rakenteen ja parametrien esittämiseen kuluu informaatiota. Tavoitteena on etsiä sellainen kompromissi, jossa rakenteen ja pakatun datan esittämiseen tarvitaan yhteensä mahdollisimman vähän bittejä. Kokonaisuutena on kompromissi verkon monimutkaisuuden ja esitystarkkuuden välillä.

Toinen vaihtoehto on bayesiläinen mitta, jota tässäkin työssä käytetään. Se perustuu verkkorakenteen posterioritodennäköisyyteen eli rakenteen todennäköisyyteen havaintojen tekemisen jälkeen. Mitan maksimoi verkko, joka sisältää etukäteisoletusten ja tehtyjen havaintojen mielessä todennäköisimmät muuttujien riippuvuudet.

Havaintojoukko $D = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(K)}\}$ koostuu K:sta näytevektorista $\boldsymbol{x}$, joiden alkioina x_i ovat havaitut arvot jokaiselle verkon N:lle satunnaismuuttujalle. Geenien ilmentymisen tapauksessa muuttujat ovat geenien ilmentymistasoja ja näytevektorit eri mittauksia. Yksittäisen näytevektorin todennäköisyys olla peräisin verkon koodaamasta jakaumasta, eli *uskottavuusfunktion* arvo, voidaan laskea ketjusäännön avulla tulona muuttujien paikallisista todennäköisyyksistä:

$$p(\boldsymbol{x}|\mathcal{S}, \boldsymbol{\theta}) = \prod_{i=1}^{N} p(x_i|\boldsymbol{u}_i, \boldsymbol{\theta}_i).$$

Olettamalla havainnot toisistaan riippumattomiksi ja keskenään samalla tavalla jakautuneiksi voidaan verkon $\mathcal{B}$ uskottavuus koko aineistolle kirjoittaa tulona yksittäisillä havaintovektoreilla $\boldsymbol{x}^{(j)}$ lasketuista uskottavuuksista:

$$p(D|\mathcal{B}) = p(D|\mathcal{S}, \boldsymbol{\theta}) = \prod_{j=1}^{K} p(\boldsymbol{x}^{(j)}|\mathcal{S}, \boldsymbol{\theta}) = \prod_{j=1}^{K} \prod_{i=1}^{N} p(x_i^{(j)}|\boldsymbol{u}_i, \boldsymbol{\theta}_i), \qquad (1)$$

missä jokainen tulon termi on edelleen purettu paikallisten todennäköisyyksien tuloksi edellisellä kaavalla.

Bayesin säännöllä Bayes-verkon rakenteen $\mathcal{S}$ posterioriksi, kun on tehty havainnot D, saadaan

$$p(\mathcal{S}|D) = \frac{p(D|\mathcal{S})p(\mathcal{S})}{p(D)}.$$

Oikean puolen osoittajan ensimmäinen termi on verkkorakenteen $\mathcal{S}$ uskottavuus, eli todennäköisyys jolla havainnot ovat peräisin annetusta verkosta. Toinen termi on rakenteen prioritodennäköisyys ja nimittäjä datan evidenssi, joka normalisoi posteriorin siten, että integraali kaikkien mahdollisten arvojen yli on yksi. Koska verkkorakenne ei vaikuta evidenssiin, voidaan evidenssi jättää huomiotta vertailtaessa verkkoja.

Usein käytetään posteriorin logaritmiä, koska sen avulla tulomuotoinen lauseke muuttuu summaksi, jota on helpompi käsitellä. Koska logaritmi on aidosti kasvava funktio, saavuttaa logaritminen posteriori maksimiarvon samassa pisteessä kuin normaali posteriorikin, joten logaritmin ottaminen ei vaikuta optimointitehtävän ratkaisuun. Verkon bayesiläiseksi yhteensopivuusmitaksi määritellään posteriorin logaritmi ilman evidenssiä, joka on sama kaikille verkoille:

$$\begin{aligned} score(\mathcal{S}|D) &= \log p(D|\mathcal{S}) + \log p(\mathcal{S}) \\ &= \log \int p(D|\mathcal{S}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{S}) \mathrm{d}\boldsymbol{\theta} + \log p(\mathcal{S}). \end{aligned} \qquad (2)$$

Alemmalla rivillä on kirjoitettu eksplisiittisesti näkyviin miten rakenneuskottavuus $p(D|\mathcal{S})$ lasketaan käyttäen lauseketta (1), joka riippuu paikallisten jakaumien parametreistä $\boldsymbol{\theta}$. Termi $p(\boldsymbol{\theta}|\mathcal{S})$ on jakaumien parametrien priori. Sen täytyy riippua verkon rakenteesta, koska solmujen paikallisten jakaumien parametrien määrä riippuu isäsolmujen määrästä. Parametrivektori $\boldsymbol{\theta}$ on tässä tapauksessa haittaparametri; sen arvot eivät ole varsinaisesti kiinnostavia lopputuloksen kannata, mutta ilman sitä termiä $p(D|\mathcal{S})$ ei ole mahdollista laskea. Bayesiläisen teorian mukaan ylläolevan kaltainen integrointi on oikea tapa päästä eroon haittaparametreistä, koska se ei valitse yhtä ainoaa arvoa vektorille $\boldsymbol{\theta}$ vaan huomioi kaikki mahdolliset arvot painottaen niitä priorijakauman mukaisesti. Jatkossa osoitetaan, että integraali on mahdollista laskea analyyttisesti tietyillä oletuksilla.

Integraalin voi myös käsittää havaintojen todennäköisyyden keskiarvon laskemiseksi parametrien priorijakauman yli. Bayesiläinen yhteensopivuusmitta siis sisältää automaattisen rankaisun monimutkaisille malleille, joille vain harvat parametrien arvot ovat hyviä, ja suosii malleja, jotka pystyvät esittämään havainnot pienemmällä määrällä parametrejä. Samalla tavalla MDL tekee kompromissin mallin monimutkaisuuden ja opetusnäytteiden esittämiseen tarvittavien bittien määrän välillä. Molemmat siis ehkäisevät ylisovittumista. Voidaankin osoittaa [14], että MDL ja bayesiläinen mitta ovat asymptoottisesti ekvivalentteja ja oikeita, eli ne molemmat suosivat todellista generoivaa jakaumaa esittävää Bayes-verkkoa, kun havaintojen määrä on tarpeeksi suuri.

Bayesiläinen Dirichlet -metriikka multinomijakautuneelle datalle

Jatkossa oletetaan, että muuttujat ovat diskreettiarvoisia ja multinomijakautuneita. Koska jokaisella muuttujalla on vain äärellinen määrä mahdollisia arvoja, voidaan jokaisen solmun isäsolmujen kaikki mahdolliset *konfiguraatiot* listata.

Multinomijakaumaa (kuten muitakin eksponenttiperheeseen kuuluvia jakaumia) käytettäessä havaintovektorien sisältämä jakauman kannalta oleellinen informaatio voidaan tiivistää niinsanotuiksi *tyhjentäviksi tunnusluvuiksi*. Kaikki jakaumista laskettavat todennäköisyydet voidaan lausua pelkästään niiden avulla eikä varsinaisia havaintovektoreita tarvitse säilyttää. Multinomijakauman tyhjentävät tunnusluvut ovat lukumääriä, jotka kertovat kuinka monta kertaa muuttuja on havaittu tietyssä tilassa. Jos multinomijakaumia käytetään Bayes-verkon paikallisina jakaumina, niin tunnuslukuja voidaan merkitä N_{ijk}:lla, jotka kertovat kuinka monta kertaa muuttuja i on havaittu tilassa k samaan aikaan kun isäsolmut ovat olleet konfiguraatiossa j. Nämä lukumäärät riittävät määräämään jakauman, koska oleellista ei ole monennessako havaintovektorissa tietty solmun ja isäsolmun arvojen yhdistelmä on esiintynyt, riittää tietää montako kertaa kyseinen konfiguraatio on havaittu.

Multinomijakauman konjugaattipriori on Dirichlet-jakauma. Se tarkoittaa, että multinomijakautuneen uskottavuuden ja Dirichlet-jakautuneen priorin tulo

on edelleen Dirichlet-jakautunut posteriori, jonka parametrit voidaan laskea suljetussa muodossa. Jos käytettäisiin jotain muuta kuin konjugaattiprioria, täytyisi posterioria approksimoida numeerisesti. Jos Dirichlet-jakauman parametrejä merkitään α_k:lla, niin Dirichlet-jakauman tiheysfunktio on

$$Dir(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{k\in \mathrm{Val}(X)} \alpha_k)}{\prod_{k\in \mathrm{Val}(X)} \Gamma(\alpha_k)} \prod_{k\in \mathrm{Val}(X)} \theta_k^{\alpha_k - 1}$$

Cooper ja Herskovits [11] johtivat ensimmäisenä analyyttisen lausekkeen yhteensopivuusmitan (2) integraalille olettamalla Dirichlet-priorin. Jos solmun X_i arvoon k, kun isäsolmut ovat konfiguraatiossa j, liittyviä hyperparametrejä (erotuksena $\boldsymbol{\theta}$-parametreistä) merkitään α_{ijk}:lla, niin integraalin logaritmin arvo on

$$\log p(D|\mathcal{S}) = \log \int p(D|\mathcal{S}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{S}) \mathrm{d}\theta = \sum_{i=1}^{N} \sum_{j\in \mathrm{Val}(\boldsymbol{U}_i)} \Phi(i,j), \tag{3}$$

missä on käytetty lyhennysmerkintää

$$\Phi(i,j) = \log \frac{\Gamma(\sum_{k\in \mathrm{Val}(X_i)} \alpha_{ijk})}{\Gamma(\sum_{k\in \mathrm{Val}(X_i)} \alpha_{ijk} + \sum_{k\in \mathrm{Val}(X_i)} N_{ijk})} + \sum_{k\in \mathrm{Val}(X_i)} \log \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}.$$

Huomionarvoista integraalissa on se, että se on summa yksittäisiin solmuihin liittyvistä termeistä. Jos lisäksi rakennepriorin logaritmi jakautuu samalla tavalla osiin, jotka riippuvat vain yhden solmun isäsolmuista $\log p(\mathcal{S}) = \sum_i \rho_i(\boldsymbol{U}_i)$, niin koko yhteensopivuusmitta (2) voidaan kirjoittaa summana yksittäisistä solmuista riippuvista termeistä:

$$score(\mathcal{S}|D) = \sum_{i=1}^{N} score_i(D, \boldsymbol{U}_i),$$

missä $score_i(D, \boldsymbol{U}_i)$ on solmun i paikallinen mitta:

$$score_i(D, \boldsymbol{U}_i) = \sum_{j\in \mathrm{Val}(\boldsymbol{U}_i)} \Phi(i,j) + \rho_i(\boldsymbol{U}_i).$$

Tästä ominaisuudesta on hyötyä rakennehaussa, kun verkkoa muutettaessa tarvitsee laskea uudelleen vain niiden solmujen, joita muutos koski, paikalliset mitat. Muutos ei vaikuta muiden solmujen osuuteen, jolle voidaan käyttää aikaisemmin laskettua arvoa, mikä on huomattavasti nopeampaa kuin koko mitan laskeminen uudelleen. Heckerman [22] kutsuu tällaista multinomiarvoisten muuttujien ja Dirichlet-priorin avulla muodostettua funktiota bayesiläiseksi Dirichlet-metriikaksi, eli BD-metriikaksi.

Hyperparametrit α_{ijk} kertovat käyttäjän oletukset verkkojen paikallisten jakaumien parametreistä ennen havaintojen tekoa. Koska hyperparametrejä on

hyvin paljon, niille ei yleensä aseteta arvoja erikseen vaan ne kootaan jollain perusteella ryhmiin, joissa kaikille hyperparametreille asetetaan sama arvo. Yksinkertaisin tapa on asettaa kaikki ykköseksi $\alpha_{ijk} = 1$, mikä tekee kaikista $\boldsymbol{\theta}$-parametrien arvoista etukäteen yhtä todennäköisiä. Buntine [7] ehdottaa sijoitusta

$$\alpha_{ijk} = \frac{\tilde{N}}{|\mathrm{Val}(X_i)||\mathrm{Val}(\boldsymbol{U}_i)|},$$

minkä voidaan osoittaa antavan yhtä suuren painon kaikille samaan ekvivalenssiluokkaan kuuluville verkoille, eli verkoille jotka koodaavat saman globalin todennäköisyysjakauman, vaikka joidenkin kaarien suunta voikin olla eri. Parametri $\tilde{N}$ mittaa käyttäjän luottamusta prioriin. Sille on intuitiivinen tulkinta kuvitteellisten näytteiden lukumääränä [22]. Priorin voi ajatella kuvaavan tietämystä, joka on syntynyt kun on jo havaittu $\tilde{N}$ kuvitteellista näytettä. Jos $\tilde{N}$ on pieni verrattuna todellisten havaintojen lukumäärään N, niin priorin vaikutus on vähäinen, jos taas $\tilde{N}$ on suuri, niin posteriori määräytyy lähes täysin priorin perusteella.

Rakenneprioreja

Rakennepriori $p(\mathcal{S})$ bayesiläisessä yhteensopivuusmitassa (2) kuvaa käyttäjän tietoa verkon rakenteesta ennen havaintojen tekoa. Yksinkertaisimmillaan se voi olla tasajakauma, jolloin kaikki verkkorakenteet ovat etukäteen yhtä todennäköisiä. Friedman [16] on ehdottanut hieman monimutkaisempaa, mutta jossain tapauksissa järkevämpää prioria. Siinä oletetaan, että solmun X_i isäsolmut $\boldsymbol{U}_i$ on valittu tasajakautuneesti kaikista vastaavan kokoisten isäsolmujoukkojen joukosta, joita on $\binom{N-1}{|\boldsymbol{U}_i|}$ kappaletta. Tietyn rakenteen prioritodennäköisyys on silloin

$$p_{\mathrm{Friedman}}(\mathcal{S}) \propto \prod_{i=1}^{N} \binom{N-1}{|\boldsymbol{U}_i|}^{-1}$$

Tämä priori suosii rakenteita, joissa solmuilla on vain muutamia isäsolmuja.

Heckerman et al. [23] ehdottavat toisenlaista prioria, joka muodostetaan vertaamalla rakennetta asiantuntijan muodostamaan verkkoon. Rakennepriori riippuu heidän mallissaan tutkittavan verkon puuttuvien ja ylimääräisten kaarien lukumäärästä δ verrattuna asiantuntijan verkkoon:

$$p_{\mathrm{Heckerman}}(\mathcal{S}) \propto \kappa^{\delta},$$

missä $0 < \kappa \leq 1$ on priorin jyrkkyyden määrävä vakio.

Kaikki edelliset ovat pehmeitä prioreita, eli ne ovat nollaa suurempia kaikille rakenteille. On myös mahdollista käyttää kovia prioreita sulkemalla vaihtoehtojen joukosta kokonaan pois joitain rakenteita. Useat tutkijat [1, 26, 38] ovat rajoittaneet solmuun tulevien kaarien määrän tiettyä rajaa pienemmäksi. Tämä on varsin tehokas rajoitus, koska esimerkiksi multinomijakaumissa parametrien määrä kasvaa eksponentiaalisesti suhteessa isäsolmujen määrään.

Kieltämällä liian tiheästi kytketyt verkot, jää parametrejä estimoitavaksi hallittavissa oleva määrä.

Haettaessa yhteensopivuusmitan maksimoivaa verkkoa, rakennepriorin ei tarvitse olla normeerattu sillä normeerausvakio on joka tapauksessa sama kaikille verkkorakenteille. Koska normeerausvakio ei vaikuta parhaan rakenteen etsintään, se voidaan yhtä hyvin jättää kokonaan pois.

Ahne optimointi Bayes-verkoissa

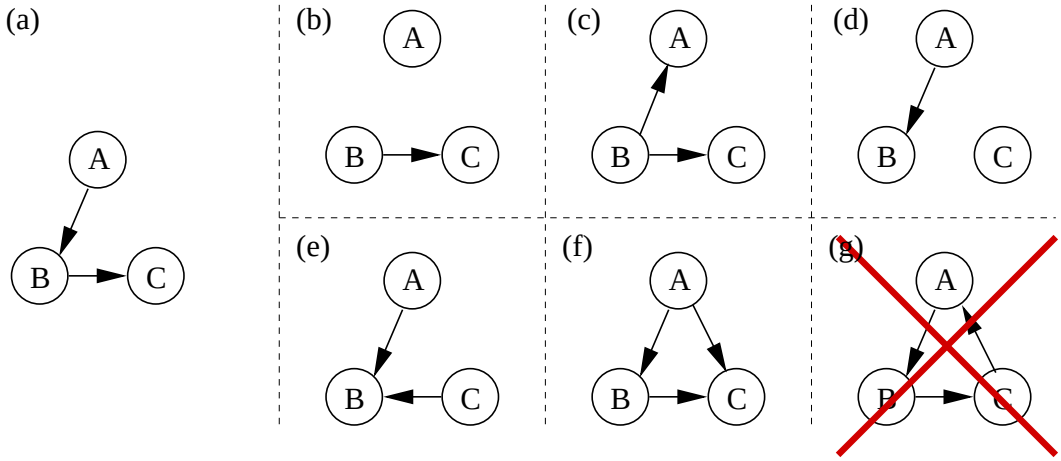
Parhaan Bayes-verkon etsintä voidaan nähdä diskreettinä optimointitehtävänä. Tavoitteena on etsiä käytettävän yhteensopivuusmitan mielessä hyvä verkko. Parhaimman mahdollisen verkon taatusti löytävä täydellinen haku on yleensä liian raskas, joten usein käytetään heuristisia algoritmejä. Ne ovat täydellistä hakua nopeampia mutta tyytyvät epäoptimaaliseen ratkaisuun.

Diskreetin optimointitehtävän tavoitteena on löytää se muuttujan x arvo, jolla kohdefunktio $f(x)$ saavuttaa maksiminsa (tai, tehtävän muotoilusta riippuen, miniminsä). Mahdollisia muuttujan arvoja voi olla hyvin paljon, mutta kuitenkin äärellinen määrä, koska kyseessä on diskreetti tehtävä. *Ahne haku* tai *ahne optimointi* on ehkä yleisemmin käytetty heuristinen optimointialgoritmi. Sitä käytetään, kun muuttujalla on liian paljon arvoja, jotta ne voitaisiin käydä yksitellen läpi ja valita niistä paras. Ahneen haun idena on valita yritteiksi $\{\hat{x}_i\}$ kaikki vanhan tilan x_i seuraajat (tilan ja seuraajien tarkempi määrittely riippuu sovelluskohteesta). Funktion $f(x)$ arvo lasketaan kaikille yritteille ja niistä hyväksytään uudeksi tilaksi x_{i+1} se joka eniten kasvattaa funktion arvoa verrattuna vanhaan tilaan x_i. Tätä toistetaan kunnes millään yritteellä ei saavuteta parempaa arvoa, kuin edellisessä tilassa, jolloin algoritmin suoritus päättyy. Selvästi iteraatio suppenee paikalliseen maksimiin (tai satulapisteeseen), koska joka iteraatiolla funktion arvo voi ainostaan kasvaa.

Nimitys *ahne optimointi* tulee siitä, että joka askeleella hyväksytään suurimman parannuksen aikaansaava uusi tila, mutta algoritmi ei yritäkään selvittää onko valinta kokonaisuuden kannalta paras. Joskus hieman huonomman tilan valinta voisi tulevaisuudessa avata uusia polkuja, jotka johtaisivat parempaan lopputulokseen. Ahneen optimoinnin ideana on laskea nopeasti vain yhden askeleen muutokset eikä huomioida tulevia askelia. Ei ole olemassa oikeastaan mitään keinoa arvioida kuinka lähellä oikeaa optimia löydetty rakaisu on.

Eräs keino paikallisesta optimista karkaamiseen on haun uudelleenkäynnistys. Ahneen optimoinnin palautettua löytämänsä optimitilan tehdään siihen muutama satunnainen muutos ja käynnistetään ahne haku uudelleen. On mahdollista, että tällä kertaa haku päätyy johonkin toiseen paikalliseen optimiin. Jos funktion arvo tässä pisteessä on suurempi kuin ensimmäisen haun jälkeen, palautetaan uusi tila, muussa tapauksessa ensimmäisen haun tulos. Muutaman uudelleenkäynnistyksen jälkeen ollaan mahdollisesti päädytty parempaan tilaan kuin vain yhden ahneen haun jälkeen.

Bayes-verkon rakenteen hakuun sovellettuna [22] tila x on ehdokasverkon rakenne ja sen seuraajat saadaan lisäämällä tai poistamalla verkosta yksi kaari tai kääntämällä yhden kaareen suunta, mutta vain jos nämä muutokset eivät tee verkosta syklistä (kuva 5). Alkutila voi olla esimerkiksi tyhjä verkko, jossa ei ole yhtään kaarta. Bayes-verkkojen ahne optimointi on erityisen kätevää, kun yhteensopivuusmitta $f(x)$ jakautuu osiin solmuittain, kuten BD-funktio. Tällöin seuraajia muodostettaessa tarvitsee laskea uudelleen vain muutettuun kaareen liittyvien solmujen osuus eikä koko funktiota.



Kuva 5: (a) Yksinkertainen Bayes-verkko. (b) – (f) Kaikki sallitut yhden kaaren poistamisella, lisäämisellä tai suunnan vaihtamisella kuvan a verkosta muodostettavat verkot. (g) Kaaren lisääminen solmusta C solmuun A ei ole sallittua, koska se tekisi verkosta syklisen.

Verkon syklisyyden testaaminen

Seuraavassa esitellään eräs algoritmi verkon syklisyyden testaamiseksi. Bayes-verkot eivät saa olla syklisiä ja siksi rakennehakualgoritmien, jotka etenevät tekemällä rakenteeseen pieniä muutoksia, täytyy pystyä varmistamaan, että muutos ei tee verkosta syklistä.

Suunnatun verkon syklittömyys voidaan testata yrittämällä muodostaa solmuille topologinen järjestys, eli yrittämällä luetella solmut sellaisessa järjestyksessä, että kaikki solmun edeltäjät (kaikki isäsolmut verkon juureen asti) esiintyvät listassa ennen itse solmua ja kukin solmu esiintyy listassa täsmälleen yhden kerran. Jos verkossa on sykli, ei topologista järjestystä ole olemassa. Silloin nimittäin kahden minkä tahansa sykliin kuuluvan solmun keskinäistä topologista järjestystä ei ole määritelty, koska molemmat ovat toistensa edeltäjiä ja kummankin pitäisi siksi esiintyä listalla ennen toista, mikä on selvästi mahdotonta.

Syklittömälle verkolle topologisen järjestyksen voi muodostaa algoritmilla 1. Alustetaan solmulista tyhjäksi. Valitaan solmu, jolla ei ole isäsolmuja. Syklit-

Algoritmi 1 Topologinen järjestäminen

Syöte: Järjestettävän verkossa solmut V ja kaaret E
Tuloste: Solmujen topologinen järjestys

1: $L \leftarrow \emptyset$
2: $i \leftarrow 1$
3: **while** $\{w \in V \mid \text{SaapuvatKaaret}(w) = \emptyset\} \neq \emptyset$ **do**
4: Valitse $v \in \{w \in V \mid \text{SaapuvatKaaret}(w) = \emptyset\}$
5: $V \leftarrow V \setminus v$
6: $E \leftarrow E \setminus \text{LähtevätKaaret}(v)$
7: $L_i \leftarrow v$
8: $i \leftarrow i + 1$
9: **end while**
10: Tulosta L

tömässä suunnatussa verkossa on aina vähintään yksi tällainen solmu[3]. Poistetaan valittu solmu ja siitä lähtevät kaaret verkosta ja lisätään solmu listan loppuun. Tätä toistetaan, kunnes verkossa ei enää ole isättömiä solmuja. Jos nyt kaikki verkon solmut on poistettu, niin solmulista on topologinen järjestys. Jos verkkoon jää käsittelemättömiä solmuja, alkuperäisessä verkossa on vähintään yksi sykli. On huomattava, että kaikki jäljelle jääneet solmut eivät välttämättä ole osa sykliä.

Eräs yleisen verkon esittämiseen käytettävä tietorakenne on nimeltään seuraajalista. Siinä jokaista verkon solmua kohden muodostetaan lista, joka sisältää niiden solmujen nimet, joihin ensimmäisestä solmusta on suora suunnattu yhteys. Jos verkko esitetään seuraajalistojen avulla, voidaan edellä kuvattu algoritmi toteuttaa lineaarisessa ajassa solmujen määrän $|V|$ ja kaarien määrän $|E|$ suhteen. Aluksi lasketaan jokaisen solmun sisääntulevien kaarien määrä käymällä läpi kaikki kaaret ja kasvattamalla jokaisen kohdalla lähtösolmun laskuria yhdellä. Tämän vie ajan $\mathcal{O}(|E|)$. Isättömien solmujen, eli niiden joihin ei tule yhtään kaarta, indeksit laitetaan pinoon (solmujen läpikäynti ajassa $\mathcal{O}(|V|)$). Varsinainen laskenta tapahtuu poistamalla pinon päällimmäinen alkio, pienentämällä alkion seuraajalistassa olevien solmujen sisääntulevien kaarien määrää yhdellä ja, jos määrä putosi nollaan, lisäämällä solmut pinoon. Tätä toistetaan kunnes pino on tyhjä. Pahimmassa tapauksessa (silloin, kun verkko on syklitön) joudutaan käymään kaikki verkon $\mathcal{O}(|V|)$ solmua lävitse. Yhteensä topologisen järjestämisen aikavaativuus on $\mathcal{O}(|E| + |V|)$. Syklittömyys on helppo todeta pitämällä kirjaa siitä montako solmua verkosta on poistettu, eli montako kertaa pinosta on otettu uusi alkio. Jos tämä laskurin arvo on lopussa sama kuin alkuperäisen verkon solmujen lukumäärä, niin algoritmi vieraili kaikissa solmuissa, muussa tapauksessa osa solmuista jäi jäljelle ja verkko on siis syklinen.

[3] Lähtemällä liikkeelle mistä tahansa solmusta ja kulkemalla päinvastaiseen suuntaan kuin kaaret osoittavat päädytään aina lopulta isättömään solmuun, koska verkko on äärellinen ja siinä ei ole syklejä.

3.2.3 Paikallisten todennäköisyysjakaumien säännöllisyyksien huomioiminen

Bayes-verkon rakenne kuvaa muuttujien välisiä globaaleja riippumattomuuksia, jotka pätevät kaikilla muuttujien arvoilla. Muuttujien välillä voi olla myös kontekstikohtaisia riippumattomuuksia. Tässä konteksti tarkoittaa jotain tiettyä muuttujien arvojen kombinaatiota tai isäsolmujen konfiguraatiota. Kaksi muuttujaa voivat olla riippumattomia, kun kolmas muuttuja on tietyssä tilassa, mutta riippuvia, kun kolmas muuttuja on toisessa tilassa. Boutilier et al. [6] määrittelevät muuttujat $\boldsymbol{X}$ ja $\boldsymbol{Y}$ *kontekstuaalisesti riippumattomiksi* annettuna $\boldsymbol{Z}$ ja konteksti $\boldsymbol{k} \in \mathrm{Val}(\boldsymbol{K})$, jos

$$p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{k}, \boldsymbol{Y}) = p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{k}), \text{ kun } p(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{k}) > 0.$$

Riittää siis, että muuttujat ovat riippumattomia vain tietyllä $\boldsymbol{K}$:n arvolla. Tällaista ominaisuutta tarvitaan olosuhderiippuvan geenisäätelyn mallintamisessa. Säätelijät voivat vaikuttaa vain joissain olosuhteissa, eli ne ovat kontekstuaalisesti riippumattomia luokkamuuttujan määräämässä kontekstissa. Tätä ajatusta käsitellään enemmän luvussa 4.

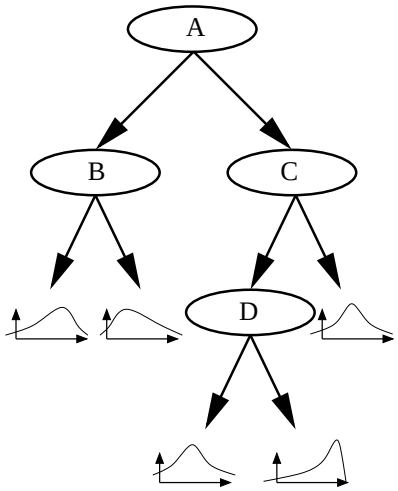
Kontekstikohtaiset riippumattomuudet näkyvät muuttujien paikallisten jakaumien todennäköisyysjakaumissa säännöllisyyksinä, koska osassa isäsolmujen konfiguraatioista muuttuja noudattaa samaa jakaumaa ja siksi osa parametreistä on samoja. Tällaisten säännöllisyyksien eksplisiittinen esittäminen johtaa tehokkaammin toimiviin opetus- ja päättelyalgoritmeihin [6].

Friedman ja Goldszmidt [14] käyttävät päätöspuita Bayes-verkon paikallisten jakaumien säännöllisyyksien esittämiseen. Jokaiseen verkon solmuun liitetään useita vaihtoehtoisia jakaumia ja päätöspuu, joka kertoo mikä jakaumista valitaan kun on havaittu isäsolmujen arvot. He näyttävät, miten päätöspuut pitää huomioida BD-mitan lausekkeessa, ja esittelevät menetelmän, joka oppii jakaumien päätöspuuesityksen Bayes-verkon rakenteenoppimisen yhteydessä.

Päätöspuut koostuvat puusolmuista[4], joihin jokaiseen liittyy yksi päätösmuuttuja, jonka jokaista arvoa vastaa yksi lapsipuusolmu (joskus osa arvoista voidaan myös niputtaa osoittamaan samaan lapsisolmuun). Kukin muuttuja saa esiintyä korkeintaan kerran jokaisella polulla juuresta lehteen ja jokaiseen lehteen liittyy todennäköisyysjakauma. Kuvassa 6 on esimerkki päätöspuusta.

Bayes-verkon paikallisia jakaumia esitettäessä päätösmuuttujat ovat Bayes-verkon isäsolmuja ja tietyn isäsolmukonfiguraation aikaansaama jakauma selviää kulkemalla puun juuresta lehtiä kohti ja siirtymällä aina siihen lapseen, johon puusolmuun liittyvän muuttujan arvo viittaa. Osa isäsolmukonfiguraatioista voi johtaa samaan jakaumaan, jos jokaisen polun varrella ei esiinny kaikkia isäsolmuja, eli jos puu ei ole täydellinen.

[4]Sekä Bayes-verkon että päätöspuiden alkioita nimitetään yleensä solmuiksi. Tässä päätöspuiden alkioita kutsutaan puusolmuiksi erotukseksi verkon solmuista.



Kuva 6: Päätöspuu. Tässä esimerkissä jokaisella muuttujista A, B, C ja D on kaksi mahdollista arvoa, joiden perusteella valitaan joko oikea tai vasen haara. Kuhunkin lehtisolmuun liittyy omanlaisensa todennäköisyysjakauma.

Chickering, Heckerman ja Meek [9] käyttävät paikallisten jakaumien esittämiseen päätösverkkoja, jotka ovat päätöspuiden yleistyksiä. Päätösverkoissa lehtisolmulla saa olla useampi kuin yksi isäsolmu. Ne pystyvät esittämään mielivaltaisen kontekstiriippuvuuden, koska samaan jakaumaan viittaavien isäsolmukonfiguraatioiden lehdet voidaan aina sulauttaa yhdeksi. Artikkelissa raportuiduissa kokeissa päätösverkkojen käyttäminen osoittautuu huomattavasti paremmaksi posterioritodennäköisyydellä mitattuna kuin päätöspuut, jotka puolestaan voittavat selvästi täysiä todennäköisyystauluja käyttävän mallin, joka ei huomioi kontekstikohtaisia riippumattomuuksia millään tavalla.

3.3 Sovelluksia säätelyverkkoihin

Tässä luvussa tutustutaan joihinkin kirjallisuudessa esiteltyihin Bayes-verkkovariaatioihin. Esiteltävät mallit pyrkivät huomioimaan erityisesti säätelyverkkojen etsimiselle tyypillisiä ongelmia, kuten opetusnäytteiden vähyys (tyypillisesti vain muutamia kymmeniä ekspressiomittauksia) ja muuttujien suuri määrä (jopa tuhansia geenejä). Jos näitä erityispiirteitä ei huomioida, vaarana on ylisovittuminen. Luonnollisesti esiteltävät mallit ovat yleiskäyttöisiä, eli ne soveltuvat säätelyn mallintamisen lisäksi muihinkin ongelmiin, joissa havaintojen määrän suhde muuttujien määrään on pieni.

Aliluvussa 3.3.1 esiteltävä verkkorakenne pakottaa solmujen paikalliset todennäköisyysfunktiot jakamaan parametrejä keskenään ja vähentää näin estimoitavia parametrejä. Luvun 3.3.2 algoritmi sallii vain tietyn määrän isäsolmuja koko verkossa jolloin ylisovittumisen riski pienenee.

3.3.1 Moduuliverkot

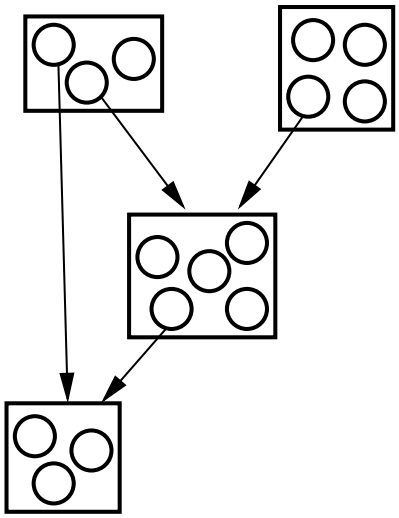
Suurimpia ongelmia Bayes-verkkojen käyttämisessä geenien säätelyverkkojen mallintamiseen on valtava parametrien määrä ja epäsäännöllinen ja siten vaikeasti tulkittava rakenne. Jokaiselle geenille tarvitaan paikallinen todennäköisyysjakauma, jonka parametrien määrä kasvaa eksponentiaalisesti suhteessa isäsolmujen määrään. Ilmentymisprofiileja on yleensä käytettävänä useista sadoista geenistä koostuvan verkon opetukseen vain joitain kymmeniä, mikä johtaa siihen, että verkko oppii pienestä näytemäärästä näennäisiä riippuvuuksia, jotka eivät kerro todellisesta säätelystä.

Ylisovittumista voi estää rajoittamalla malliperhettä. Rajoitus voidaan asettaa esimerkiksi paikallisten jakaumien muodolle [6, 21] tai geenikohtaisten säätelijöiden lukumäärälle. Eräs yleiskäyttöinen idea on parametrien jakaminen, eli samojen parametrien käyttäminen eri solmujen paikallisissa jakaumissa, jolloin jakaumat riippuvat osittain toisistaan. Parametrien jakamista on sovellettu monissa todennäköisyysmallinnuksen ongelmissa. Sen etuna on, että estimaatit parametrien arvoille saadaan laskettua suuremmasta joukosta näytteitä (kaikista jotka jakavat saman parametrin), jolloin estimaatti on luotettavampi, ja estimoitavien parametrien kokonaismäärä pienenee. Vaarana on, että jos parametrejä jaetaan liian monen jakauman kesken, mallista tulee liian jäykkä eikä sitä pystytä sovittamaan havaintoihin.

Moduuliverkko [43, 44] on erityinen Bayes-verkkorakenne, jossa samoin käyttäytyvät solmut jaetaan automaattisesti moduuleihin (kuva 7). Yhden moduulien sisällä kaikki solmut noudattavat yhteistä todennäköisyysjakaumaa ja niiden arvot siis riippuvat samoista isäsolmuista samalla tavoin. Tällainen modulaarinen rakenne on erityisen käytännöllinen silloin kun mallinnettavassa ilmiössä muuttujaryhmät toimivat koordinoidusti. Näin on esimerkiksi geenisäätelyssä, jossa muutama säätelytekijä vaikuttaa suuren geenijoukon ilmentymiseen. Opittua moduuliverkkoa on helpompi tulkita kokonaisten biologisten prosessien tasolla, kuin normaalia Bayes-verkkoa. Toinen artikkelissa käytetty esimerkki on osakemarkkinat, missä osakkeiden hintojen voi olettaa muuttuvan markkina-aloittain samansuuntaisesti.

Moduuliverkko on matemaattisesti hyvin samankaltainen kuin tavallinen Bayes-verkko. Moduuliverkon $\mathcal{M}$ rakenne $\mathcal{S}$ määrittelee isäsolmut ja parametrivektori $\boldsymbol{\theta}$ paikallisten jakaumien parametrit jokaiselle moduulille yksittäisten solmujen sijaan. Näiden lisäksi tarvitaan kolmaskin komponentti: solmujen sijoittelu moduuleihin. Sijoittelufunktio $\mathcal{A}$ liittää jokaisen solmun i johonkin moduuliin m, $\mathcal{A}(i) = m$. Vain sellaisia solmuja, joiden arvojoukko on sama voi yhdistää samaan moduuliin, mikä ei tietenkään ole ongelma, jos kaikilla muuttujilla on sama arvoalue. Samaan moduuliin sijoitettujen solmujen havaintojen ajatellaan olevan toistoja moduulin muodollisesta muuttujasta $\mathcal{M}_i$. Moduulien lukumäärä täytyy kiinnittää etukäteen ja se on seuraavaksi esiteltävässä rakennehakumenetelmässä vakio.

Koska moduuliverkko on rajoitettu versio tavallisesta Bayes-verkosta, onnistuu



Kuva 7: Yksinkertainen moduuliverkko. Selvyyden vuoksi kuvaan on piirretty vain yhdet nuolet isäsolmujen ja lapsimoduulien välille, mutta todellisuudessa isäsolmu vaikuttaa kaikkiin lapsimoduulin solmuihin.

oppiminen lähes samalla tavoin optimoimalla. BD-funktio voidaan helposti laajentaa moduuliverkoille. Moduuliverkon rakenteen ja sijoittelun posteriorin logaritmiksi tulee Bayesin kaavan ja $\boldsymbol{\theta}$-parametrin integroinnin avulla (vertaa yhteensopivuusmittaan (2))

$$score(\mathcal{A}, \mathcal{S}|D) = \log \int p(D|\mathcal{A}, \mathcal{S}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{A}, \mathcal{S}) \mathrm{d}\boldsymbol{\theta} + \log p(\mathcal{A}) + \log p(\mathcal{S}|\mathcal{A}), \quad (4)$$

mistä on jälleen jätetty pois normeeraustekijät. Sijoittelu- ja rakennepriorit $p(\mathcal{A})$ ja $p(\mathcal{S}|\mathcal{A})$ eivät voi olla täysin riippumattomia sillä verkko ei saa olla syklinen, eli esimerkiksi sellaisen verkon, jossa moduulin A isäsolmu on sijoitettu moduulin B ja moduulin B isäsolmu moduulin A prioritodennäköisyys on oltava nolla. Siksi rakennepriori on muotoa

$$p(\mathcal{S}|\mathcal{A}) \propto \begin{cases} p(\mathcal{S}), & \text{kun } \mathcal{A} \text{ ja } S \text{ määrittelevät syklittömän verkon} \\ 0, & \text{mulloin.} \end{cases}$$

Posterioritodennäköisyyttä (3) vastaava summa moduuliverkolle on [43]

$$\log p(D|\mathcal{S}, \mathcal{A}) = \sum_{m=1}^{M} \sum_{j \in \mathrm{Val}(\boldsymbol{U}_m)} \Phi^M(m, j), \quad (5)$$

missä

$$\Phi^M(m, j) = \log \frac{\Gamma(\sum_{k \in \mathrm{Val}(M_m)} \alpha^M_{mjk})}{\Gamma(\sum_{k \in \mathrm{Val}(M_m)} \alpha^M_{mjk} + \sum_{k \in \mathrm{Val}(M_m)} N^M_{mjk})} + \sum_{k \in \mathrm{Val}(M_m)} \log \frac{\Gamma(\alpha^M_{mjk} + N^M_{mjk})}{\Gamma(\alpha^M_{mjk})}$$

ja N^M_{mjk} ovat moduulikohtaisia tyhjentäviä tunnuslukuja, jotka saadaan laskemalla yhteen kaikkien moduuliin kuuluvien solmujen tunnusluvut:

$$N^M_{mjk} = \sum_{i:\mathcal{A}(i)=m} N_{ijk}.$$

Yläindeksi M korostaa, että kyseessä on kokonaiseen moduuliin liittyvä tunnusluku. Tuloksena on kaikkien niiden havaintojen lukumäärä, joissa jokin moduuliin m kuuluva solmu on ollut tilassa k samalla kun moduulin isäsolmut ovat olleet konfiguraatiossa j. Samaan moduuliin kuuluvien solmujen havaintojen voidaan siis ajatella olevan ikäänkuin toistoja samasta kuvitteellisesta "moduulimuuttujasta". Vastaavasti moduuliverkon Dirichlet-priorin hyperparametrejä on merkitty α^M_{mjk}. Hyperparametrien vaikutus on samankaltainen kuin tavallisissa Bayes-verkoissa. Esimerkiksi kaikkien hyperparametrien asettaminen ykköseksi tekee kaikista paikallisten jakaumien parametrien θ arvoista yhtä todennäköisiä.

Moduuliverkon posteioritodennäköisyys on summa yksittäisiin moduuleihin liittyvistä termeistä. Jos lisäksi priorien logaritmit voidaan kirjoittaa samalla tavalla summina

$$\log p(\mathcal{A}) = \sum_m \kappa_m(A_m)$$

ja

$$\log p(\mathcal{S}) = \sum_m \rho_m(\boldsymbol{U}_m),$$

missä κ_m ja ρ_m ovat ei-negatiivisia funktioita ja A_m on moduuliin m kuuluvien solmujen joukko, niin yhteensopivuusmitta (4) on summa moduulien paikallisista mitoista:

$$score(\mathcal{A}, \mathcal{S}|D) = \sum_{m=1}^{M} score^M_m(A_m, \boldsymbol{U}_m, D), \tag{6}$$

ja paikalliset mitat ovat muotoa

$$score^M_m(A_m, \boldsymbol{U}_m, D) = \sum_{j \in \mathrm{Val}(\boldsymbol{U}_m)} \Phi^M(m, j) + \kappa_m(A_m) + \rho_m(\boldsymbol{U}_m). \tag{7}$$

Tätä moduulien paikallisten yhteensopivuusmittojen riippumattomuutta voidaan hyödyntää verkon oppimisessa samalla tavalla kuin aikaisemmin tehtiin tavallisten Bayes-verkkojen kohdalla päivittämällä joka iteraatiolla vain muuttuneiden moduulien osuus.

Moduuliverkko optimoidaan ahneella haulla. Riippuvuusrakenteen etsimisen lisäksi nyt tulee hakea solmujen sijoittelu moduuleihin, siksi optimointialgoritmi on kaksiosainen iteraatio. Ensimmäisessä vaiheessa haetaan rakennetta samalla tavalla kuin normaaleissa Bayes-verkoissa pisteyttämällä kaikki yhden kaaren lisäykset ja poistot (kaaren suunnan kääntäminen ei ole sallittua, koska kaari osoittaa yhdestä solmusta kaikkiin lapsimoduulin solmuihin). Ehdotetun muutoksen syklisyyden tarkistamisen voi tehdä moduulien muodostamassa verkossa, jossa jokaista moduulia vastaa yksi solmu ja solmusta A on kaari

solmuun B, jos jokin moduulin A solmuista on moduulin B isäsolmu. Se on nopeampaa kuin yksittäisten solmujen muodostaman verkon syklisyydentestaus, koska moduuleita on normaalisti paljon vähemmän kuin solmuja. Muutoksista hyväksytään eniten yhtensopivuusmitan arvoa kasvattava ja iteraatiota toistetaan kunnes ei enää löydy hyväksyttäviä yhden kaaren muutoksia.

Toisessa vaiheessa koetetaan löytää solmuille parempi sijoittelu pitäen rakennetta kiinteänä. Solmujen sijoittelua ei voi optimoida toisistaan riippumattomasti, koska verkkoon ei saa muodostua syklejä. Jos esimerkiksi solmu i on moduulin A ja solmu j moduulin B isäsolmu, niin solmua i ei voi sijoittaa moduulin B samanaikaisesti kun solmu i kuuluu moduuliin A. Toinen syy solmujen sijoittelun riippuvuuteen on, että BD-funktio on moduulien tyhjentävien tunnuslukujen suhteen epälineaarinen. Siksi on mahdollista laskea yhteensopivuusmitan muutos siirrettäessä solmu moduulista toiseen vain, jos muut solmut pysyvät samalla paikoillaan.

Sijoittelun hakukin toteutetaan ahneella optimoinnilla. Solmut käydään yksitellen läpi ja jokaista kokeillaan siirtää kaikkiin muihin moduuleihin pitäen koko muu verkko vakiona. Solmuja siirrellessä täytyy jälleen pitää huoli siitä, ettei muodostu syklejä. Jokaiselle sallitulle siirtoyritykselle lasketaan yhteensopivuusmitan muutos ja, jos se on positiivinen, siirto hyväksytään ja päivitetään verkkoon. Solmuja käydään läpi kunnes minkään solmun sijoitus ei enää muutu, jolloin sijoittelun optimointi päättyy. Löydetty ratkaisu on optimaalinen vain siinä mielessä, että yksittäisten solmujen siirtäminen ei sitä paranna, mutta se ei tarkoita sitä, etteikö useamman solmun siirtäminen kerrallaan voisi parantaa mitan arvoa.

Näitä kahta vaihetta toistetaan peräkkäin kunnes lopetuskriteeri täyttyy. Lopetus voi tapahtua esimerkiksi silloin, kun yhteensopivuusmitta ei enää muutu yhden kokonaisen iteraation aikana tai kun suhteellinen muutos on pienempi kuin jokin jokin raja-arvo. Koska kummassakaan vaiheessa sopivuusmitta ei voi pienentyä, menetelmä konvergoi paikalliseen maksimiin, mutta, kuten aina heuristisilla menetelmillä, globaalin optimin löytymistä ei voi taata.

Ennen opetuksen alkua solmujen sijoittelu täytyy alustaa jollain tavalla. Artikkelissa ehdotetaan todennäköisyyteen perustuvaa kokoavaa klusterointia. Samaan klusteriin laitettiin geenit, joiden ilmentymisprofiilit voisivat olla peräisin samasta todennäköisyysjakaumasta.

Artikkelissa opetusta testattiin generoimalla näytteitä moduuliverkosta ja opettamalla uusi verkko näytteiden perusteella. Vertaamalla opittua verkkoa todelliseen generoivaan verkkoon kirjoittajat osoittavat, että jos näytteitä on tarpeeksi, niin on mahdollista oppia todellisen verkon riippuvuudet kohtuullisella tarkkuudella.

Algoritmin testaamiseksi todellisessa tilanteessa moduuliverkko sovitettiin alunperin hiivan stressin tutkimiseksi mitattujen 173 ilmentymisprofiiliin. Opetuksesta jätettiin pois geenit, joiden ilmentymistaso ei vaihdellut kovin paljoa.

Jäljelle jäi 2355 geeniä, joista 466 on kirjallisuuden perustella potentiaalisia säätelytekijöitä. Vain niiden sallittiin olevan moduulien isäsolmuja opittavassa verkossa. Moduulien määräksi valittiin 50, koska kirjoittajien mukaan on biologisesti järkevää olettaa säädeltävän moduulin kooksi noin 50 geeniä.

Kirjoittajat vertasivat opitun verkon moduuleita tunnettuihin solun toiminnallisiin ryhmiin. Moduuleista 42/50 vastasi jotain tunnettua ryhmää paremmin kuin voitiin odottaa sattuman perusteella (p-arvo < 0.005). Toisessa julkaisussa [44] he tutkivat myös moduulien säätelijöitä ja totesivat, että menetelmä ennusti 30 moduulille ainakin yhden oikean, ennalta tunnetun säätelijän. He myös testasivat kolmea menetelmän ennustamaa ennalta tuntematonta säätelyvuorovaikutusta laboratoriokokeissa. Kahdessa tapauksessa he löysivät vahvaa näyttöä siitä, että menetelmän ennustus on todellinen biologinen vuorovaikutus ja kolmannessa tapauksessakin ennustetulla säätelijällä oli havaittavaa vaikutusta ilmentymistasoihin.

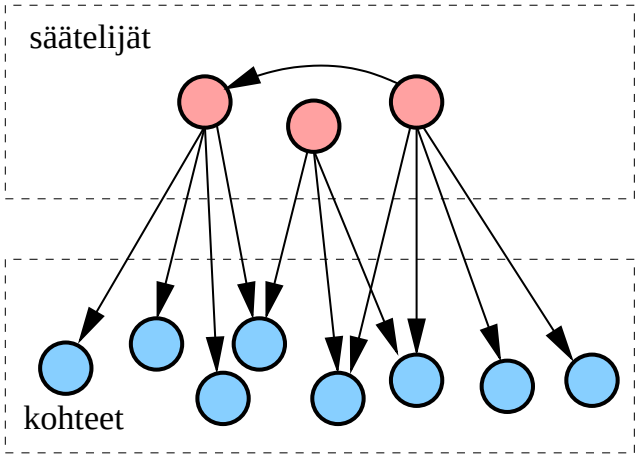
Moduuliverkon oppimisalgoritmin ongelmana voi pitää sitä, että geeni voi kuulua vain yhteen moduuliin, vaikka todellisuudessa tiedetään, että geenillä voi olla useita erilaisia tehtäviä solussa. On myös mahdollista, että jos säätelijä käyttäytyy samalla tavalla kuin kohdegeenit, algoritmi saattaa laittaa säätelijän kohdemoduuliin, jolloin säätelyvuorovaikutus jää löytämättä. Myös moduulin määrän päättäminen on vaikeaa; jos opittavassa verkossa on liian vähän moduuleita, ei kaikkia säätelijöitä löydetä, koska ne vaikuttavat vain osaan moduulin geeneistä. Jos taas moduuleita on liikaa, kärsii tulosten luotettavuus, kun estimoitavana on enemmän parametrejä.

3.3.2 MinReg-verkot

Usein käytetty keino Bayes-verkon ylisovittumisen estämiseksi on rakenteen rajoittaminen siten, että estimoitavien parametrien määrä pienenee. Pe'er et al. ovat esitelleet mallin, jossa verkon rakennetta rajoitetaan biologisesti mielekkäällä tavalla [38]. Heidän MinReg-algoritminsa on tarkoitettu etsimään tärkeimpiä säätelijöitä ja he testaavat sitä hiirestä mitatuilla ilmentymisprofiileilla. Aiemmin vain harvat menetelmät ovat toimineet tyydyttävästi nisäkkäiden monimutkaisilla säätelyverkoilla [5].

MinReg-menetelmä perustuu kolmeen biologisesti motivoituun rajoitukseen verkon rakenteelle. Ensinnäkin isäsolmujen täytyy kuulua etukäteen määrättyyn potentiaalisten säätelytekijöiden joukoon (vrt. luku 2.5), jota merkitään C:llä. Toiseksi, kaikkien vähintään yhtä geeniä säätelevien solmujen muodostaman joukon $\mathcal{R}$ koko ja jokaisen solmun isäsolmujen maksimimäärä on rajoitettu. Kirjoittajat perustelevat rajoituksia pienemmän laskentatyön lisäksi vetoamalla aikaisempiin tutkimuksiin, joiden mukaan yleensä vain pieni joukko potentiaalisista säätelytekijöistä on aktiivisia kerrallaan ja niistä kukin säätelee suurta joukkoa geenejä. Kaikkien isäsolmujen täytyy kuulua säätelijäjoukkoon $\mathcal{R}$. Verkkorakenne on siis ikään kuin kaksikerroksinen, jossa ylhäällä on

säätelijät ja alhaalla kohdegeenit (kuva 8).



Kuva 8: MinReg-verkossa vain ylempään kerrokseen kuuluvat säätelijät voivat olla isäsolmuja alakerroksen kohdegeeneille.

Rakenteen lisäksi toinen tärkeä osa MinReg-mallissa on optimointimenetelmä. Toisin kuin rajoittamattomissa Bayes-verkoissa, isäsolmuja ei voida hakea riippumattomasti jokaiselle solmulle erikseen. Jos mallia yritettäisiin optimoida ahneella optimoinnilla, kuten Bayes-verkkoja usein optimoidaan, ei todennäköisesti löydettäisi kovinkaan hyvää ratkaisua. Ahne haku lisää kaaren kahden solmun välille muista solmuista riippumatta, jos lisääminen kasvattaa yhteensopivuusmittaa. Tämä tarkoittaa, että todennäköisesti melkein joka askeleella säätelijäkerrokseen lisättäisiin uusi solmu, joka on hyvä säätelijä yhdelle geenille, ja jo muutaman askeleen jälkeen säätelijäkerros olisi täynnä eikä kaarien poistaminen todennäköisesti kasvattaisi sopivuusmittaa. Silloin sadat geenit jäisivät kokonaan ilman säätelijää, mikä ei ole kovin realistista. Siksi kirjoittajat esittelevät toisenlaisen optimointimenetelmän, joka korottaa solmun säätelijäkerrokseen vain, jos se on hyvä säätelijä monelle geenille yhtäaikaa.

Käytettävän kustannusfunktion johtamiseksi tarkastellaan aluksi yksikertaista tapausta, jossa säätelijäkerroksen geenit on kiinnitetty. Tällöin on mahdollista valita jokaiselle kohdegeenille X yhteensopivuusmitan $score(X, \boldsymbol{U})$ mielessä parhaat säätelijät yksinkertaisesti laskemalla funktion arvo kaikilla säätelijöiden osajoukoilla $\boldsymbol{U} \subset \mathcal{R}$ ja valitsemalla niistä paras. Tämä on mahdollista, jos solmuihin sisään tulevien kaarien määrän yläraja d ja säätelijäjoukon maksimikoko k ovat tarpeeksi pieniä. Kirjoittajat suosittelevat d:lle arvoja väliltä 3–5 ja k:lle väliltä 30–70. Silloin tarvitsee tutkia vain $\binom{k}{d}$ osajoukkoa. Se on hyvin paljon vähemmän kuin ilman säätelijäjoukon koon rajoitusta syntyvät $\binom{N}{d}$ osajoukkoa, koska geenien lukumäärä N voi olla useita tuhansia.

Jokaiselle geenille (sekä kohteille että säätelijöille) etsitään erikseen parhaat säätelijät maksimoimalla geenikohtaiset yhteensopivuusmitat. Maksimaalisten mittojen summa kiinteällä säätelijäjoukolla $\mathcal{R}$ on säätelijäjoukon utiliteetti-

funktio:

$$F(\boldsymbol{R}) = \sum_{i=1}^{N} \max_{\boldsymbol{P} \subset \boldsymbol{R}, |\boldsymbol{P}| \leq d} score(X_i, \boldsymbol{P}).$$

Tarkalleen ottaen $F(\boldsymbol{R})$ ei ole optimaalinen mitta $\boldsymbol{R}$:n hyvyydelle, koska se lasketaan maksimoimalla erikseen kunkin geenin säätelijät, mikä ei huomioi mahdollisesti muodostuvia syklejä. Kirjoittajat väittävät, että tarkkuus on käytännössä varsin hyvä, koska syklejä voi muodostua vain säätelijäkerrokseen, joka on hyvin pieni osa koko verkosta.

Varsinainen optimointitehtävä on hakea parhaat geenit potentiaalisten säätelytekijöiden joukosta C säätelijöiksi $\boldsymbol{R}$. Valituksi tulevat utiliteetin maksimoivat geenit:

$$\boldsymbol{\mathcal{R}} = \underset{\boldsymbol{R} \subset C, |\boldsymbol{R}| \leq k}{\arg\max}\ F(\boldsymbol{R})$$

Koska mahdollisia säätelijäjoukkoja on niin paljon, niiden läpikäynti yksitellen ei ole mahdollista. Tässä kirjoittavat turvautuvatkin ahneeseen hakuun, lisäten joka askeleella säätelijäksi sen solmun, jonka lisääminen kasvattaa utiliteettia eniten. Oleellista on, että uutta säätelijää valittaessa yritetään sovittaa se mahdollisimman monen geenin säätelijäksi kerrallaan. Silloin vältetään säätelijäkerroksen ennenaikainen täyttyminen, joka seuraisi ahneen haun käyttämisestä.

Uusia säätelijöitä lisätään malliin kunnes utiliteetin lisäys ei enää merkittävästi ylitä satunnaisen säätelijän lisäämisen vaikutusta. Oikean lopetushetken arvioimiseksi valitaan osajoukko säätelijöistä ja permutoidaan niiden havainnot, jolloin ne noudattavat samaa jakaumaa kuin todeliset säätelijät, mutta ovat riippumattomia kohdemuttujien arvoista. Laskemalla näin muodostettujen satunnaissäätelijöiden lisäyksien aiheuttamat utiliteetin muutokset saadaan selville satunnaisten säätelijöiden lisäämisen empiirinen jakauma. Algoritmi lopetetaan, kun sen valitseman säätelijän utiliteetin lisäys ei eroa merkitsevästi estimoidusta jakaumasta.

Ahneena algoritmina MinReg ei onnistu löytämään globaalia optimia, jos jollekin geenille X ja säätelijöille A ja B summa $score(X, A) + score(X, B)$ on paljon pienempi kuin $score(X, A \cup B)$. Silloin ahne haku ei valitse kumpaakaan säätelijää A tai B erikseen, eikä siksi koskaan tule testaamaan myöskään niiden unionia $A \cup B$. Tiedetään, että todellisissa säätelyverkoissa esiintyy tällaista synergiaa, siksi on mielenkiintoista, että MinRegille voidaan johtaa raja ahneesta hausta aiheutuvalle virheelle.

Virhearvio perustuu synergian mittaamiseen α-modulaarisuuden avulla. Funktio f on α-modulaarinen ($\alpha \geq 1$), jos ja vain jos kaikille osajoukoille A ja R ja alkioille Z pätee

$$f(A \cup Z|R) \leq f(A|R) + \alpha f(Z|R),$$

missä $f(A|R) = f(A \cup R) - f(R)$ on joukon A lisäämisestä aiheutuva funktion arvon kasvu. α-modulaarisuus on eräänlainen konveksisuus- tai synergiamitta uusien säätelijöiden lisäämiselle.

Artikkelissa osoitetaan, että MinReg-algoritmin löytämän ratkaisun utiliteetti F_{MINREG} on aina enintään tietyn etäisyyden päässä täydellisellä haulla saavutettavun optimaalisen ratkaisun utiliteetista F_{OPT}:

$$F_{\text{MINREG}} \geq \frac{1}{\alpha + 1} F_{\text{OPT}}.$$

Raja ei ole kovin tiukka; parhaimmillaankin, kun synergiaa ei esiinny ($\alpha = 1$), artikkelin todistus takaa ainoastaan, että MinReg löytää ratkaisun, jonka utiliteetti on puolet maksimista. Epäyhtälön johtamisen merkitys on siinä, että ahnetta optimointia käyttävälle Bayes-verkon rakenteenhaku algoritmille pystytään ensimmäistä kertaa antamaan ylipäätänsä minkäänlainen virhearvio. Aikaisemmin on vain täytynyt luottaa siihen, että haku ei jää jumiin paikalliseen optimiin kovin kauaksi todellisesta ratkaisusta. Käytännössä modulaarisuuskertoimen α arvo voidaan estimoida empiirisesti havainnoista.

Esimerkkeinä todellisesta sovelluskohteista algoritmilla opetettiin kaksi verkkoa, ensimmäinen hiivasta mitatuilla ilmentymisarvoilla ja toinen hiiren B-lymfosyyttien ilmentymisprofiileilla. Hiiva on suhteellisen tarkkaan tutkittu organismi ja tarjoaa siksi hyvän vertailukohdan. Hiiren, ja ylipäätänsä kaikkien nisäkkäiden, säätelyverkostot ovat paljon hiivan verkostoja monimutkaisempia, eivätkä aikaisemmat menetelmät käytännössä ole skaalautuneet niiden mallintamiseen.

Tulosten biologisen paikkansapitävyyden testaamiseksi kirjoittajat laskivat geeniontologialuokkien rikastumat jokaisessa säätelijöiden kohdegeenien joukossa ja assosioivat säätelijät niihin GO-luokkiin, joiden rikastuman p-arvo oli pienin. Näitä säätelijöille ennustettuja GO-termejä verrattiin kirjallisuudessa esiintyneisiin luokituksiin. Hiivan tapauksessa kahdeksan kymmenestä p-arvojen mukaan merkittävimmästä ja hiirelläkin yli puolet (45/75) kaikista säätelijöistä olikin aikasemmassa tutkimuksessa yhdistetty MinRegin ennustamaan GO-luokkaan. Tulos osoittaa, että MinReg löysi biologisesti järkeenkäypiä säätelyvuorovaikutuksia. Biologin kannalta mielenkiintoisimpia kuitenkin ovat ne säätelysuhteet, joille ei löytynyt aikaisemmasta tiedosta vahvistusta. Ne ovat hyviä jatkotutkimuksen kohteita.

Kirjoittajat vertasivat vielä MinRegin ja moduuliverkkoalgoritmin ennustamia hiiren säätelyverkkoja. Aikaisemmin oli osoitettu, että moduuliverkko toimii hyvin hiirtä yksinkertaisemmalla hiivalla. He opettivat moduuliverkon samoilla hiirinäytteillä, joita oli käytetty MinRegin opetuksessa, ja suorittavat edelläkuvatun kaltaisen geeniontologialuokka-analyysin moduuliverkon ennustamille säätelijöille. Tulokset olivat paljon huonompia kuin MinReg-menetelmällä. Vaikka molemmat menetelmät löysivät osittain samat säätelijät, MinRegin säätelijöiden ennalta tunnetuista assosiaatioista 23 oli sellaisia, joita moduuliverkko ei löytänyt, kun taas moduuliverkko päihitti MinRegin vain yhden säätelijän tapauksessa. Syyksi kirjoittajat arvelevat, että moduuliverkko on liian rajoitettu toimiakseen monimutkaisen nisäkkään säätelyverkon mallinnuksessa, koska se pakottaa samaan moduuliin kuuluvat geenit noudattamaan samaa jakaumaa. MinReg puolestaan keskittyy löytämään vain tärkeimmät säätelijät

ja estimoi yksittäisten geenien käyttäytymisen havaintojen perusteella.

3.3.3 Luonnollisenkaltaisten mittausten simulointi

Säätelyverkkojen etsintäalgoritmien tehokkuuden vertailua varten tarvitaan mittaustuloksia tunnetusta säätelyverkosta. Opettamalla eri menetelmät samoilla näytteillä ja vertaamalla lopputuloksia alkuperäiseen verkkoon saadaan selville kuinka lähellä todellista menetelmän löytämä verkko on. Tässä aliluvussa perehdytään siihen, miksi keinotekoisesti muodostetut havainnot ovat hyviä menetelmien vertailuun ja esitellään eräs keino arpoa realistisenkaltaisia säätelyverkkoja ja generoida niistä synteettisiä ilmentymisprofiileja.

Vaikka useita todellisia biologisia säätelyverkkoja on tutkittu paljonkin ja tutkimustulokset ovat vapaasti saatavilla tietokannoista, on niissä kuitenkin vielä niin paljon tuntemattomia kohtia, että aitojen ilmentymismittausten käyttäminen menetelmien oppimien verkkojen vertailussa ei tuota luotettavia tuloksia. Vaikein ongelma vain osittain tunnettuihin verkkoihin verrattaessa on se, että jos algoritmi ennustaa linkin kahden sellaisen geenin välille, joita ei tietokannan verkossa ole yhdistetty, ei voi olla varma onko tämä väärä ennustus algoritmilta vai todellinen ennaltatuntematon biologinen tosiasia. Ennen ilmentymisaineistolla kokeilemista on hyvä saada jonkinlainen käsitys menetelmän hyvyydestä simuloidulla aineistolla. Jos jokin menetelmä toimii hyvin realistisenkaltaisella aineistolla, voi sen olettaa toimivan myös todellisissa mittauksissa.

Toinen ongelma on aitojen ilmentymismittausten pieni määrä. Näytteiden tulisi olla sellaisista mittauksista, joissa tutkittava säätelyverkko on toiminut monipuolisesti erilaisissa olosuhteissa. Jos dataa on hyvin vähän, minkään menetelmän ei voi olettaa toimivan hyvin.

Simulointia varten täytyy päättää verkon riippuvuusrakenne, joka kertoo jokaisen geenin säätelytekijät, ja paikalliset siirtofunktiot, jotka määräävät kuinka voimakkaasti ja mihin suuntaan säätelysuhde vaikuttaa. Keinotekoisten säätelyverkkojen muodostamiseen on käytetty kahta toisistaan poikeavaa tapaa. Ensimmäisessä on käsin rakennettu pieni (yleensä korkeintaan 10 solmua) verkko, joka vastaa mahdollisimman tarkasti tunnettuja säätelyverkkoja, ja valittu siirtofunktioiksi biologisia tietämykseen perustuva differentiaaliyhtälösysteemi [53, 45]. Tällä tavoin voi muodostaa hyvinkin realistisia verkkoja, jos tuntee tarpeeksi tarkasti biologiset vuorovaikutukset, mutta tarvittava työmäärä kasvaa hyvin suureksi hiemankaan isommilla verkoilla. Differentiaaliyhtälöiden ratkaisuna saadaan ajasta riippuva, jatkuva-arvoinen estimaatti geenien ilmentymistasoille. Aikasarja voi kuvata esimerkiksi geenien aktiivisuuden muutoksia sen jälkeen kun solua on hieman häiritty.

Toisessa ääripäässä verkko muodostetaan arpomalla suuri, jopa tuhansista solmuista solmuista koostuva satunnaisverkko, jonka tilastolliset ominaisuudet noudattavat tämänhetkistä tietämystä säätelyverkoista [36, 28]. Tällaisia ominaisuuksia ovat mm. mittakaavattomuus ja pieni maailma -ilmiö, jota nou-

dattavissa verkoissa lähes kaikki solmut ovat lyhyen polun päässä toisistaan. Siirtofunktioina käytetään yleensä differentiaaliyhtälöitä, mutta myös diskreettiarvoisia Boolen funktioita ja Bayes-verkkoja on käytetty.

Synthetic Transcriptional Regulatory Networks – SynTReN

SynTReN (Synthetic Transcriptional Regulatory Networks) [48] on äskettäin julkaistu menetelmä säätelyverkkojen generointiin. Sen sijaan että verkko muodostettaisiin täysin satunnaisesti SynTReN-ohjelma rakentaa verkon yhdistelemällä paloja tunnetuista hiivan tai kolibaktreerin säätelyverkoista. Verkko rakennetaan iteratiivisesti arpomalla satunnainen geeni (ja haluttaessa myös sen välittömät naapurit) tietokannasta ja lisäämällä se jo muodostettuun verkkoon, jos sillä on tietokannan mukaan yhteyksiä verkossa jo oleviin geeneihin. Koska todellisista verkoista koostuvassa tietokannassa on syklejä, voi niitä tulla mukaan myös arvottuun verkkoon. Verkko ei siis kelpaa suoraan Bayes-verkon riippuvuusrakenteeksi. Ohjelma pystyy myös arpomaan siirtofunktiot solmuille ja generoimaan ilmentymisnäytteitä verkosta.

Artikkelin toinen pääkontribuutio on testi, joissa empiirisesti näytetään, että edellä kuvatulla tavalla muodostetut verkkojen tilastolliset ominaisuudet, kuten keskimääräinen polun pituus ja solmuun keskimäärin tulevien kaarien määrä, ovat paljon lähempänä tunnettuja todellisia verkkoja kuin muilla keinoin tuotetuilla satunnaisverkoilla. Muilla satunnaisverkoilla yleensä jokin ominaisuus saadaan suurinpiirtein kohdalleen mutta samaan aikaan muut ominaisuudet ovat kaukana oikeista arvoistaan.

Koska SynTReNissä verkkoja rakennetaan ottamalla osia ennalta tunnetuista säätelyverkkoista, riippuu rakenteen todenmukaisuus aiemman tutkimuksen tasosta ja on painottunut laajasti tutkittuihin säätelyverkoston osiin, mutta pienet virheet, kuten puuttuvat tai ylimääräiset yksittäiset linkit, tietokannassa eivät luultavasti vaikuta kovin paljoa lopputulokseen.

Näytteiden generointi Bayes-verkosta

Kun verkon riippuvuusrakenne ja solmujen parametrit on kiinnitetty, seuraava vaihe on näytevektorien generointi. Vektorien alkioiksi tulee kullekin solmulle generoitava arvo ja vektorien suhteellisten lukumäärien tulisi noudattaa verkon määräämää jakaumaa. Generointiprosessin yksityiskohdat riippuvat tietenkin solmujen siirtofunktioiden tyypistä. Esimerkiksi differentiaaliyhtälöjä käyttävästä verkosta saadaan näytteitä päättämällä lähtötila ja integroimalla siitä askeleittain eteenpäin. Seuraavassa esitellään tarkemmin yksinkertainen menetelmä Bayes-verkkojen arvojen simulointiin.

Aluksi käsitellään isättömät solmut. Koska niiden jakaumat eivät riipu muista solmuista, voidaan niille arpoa arvot toisistaan riippumattomasti. Arvontaprosessi riippuu paikallisen jakauman tyypistä. Esimerkiksi multinomijakaumasta

saa otettua näytteitä arpomalla tasajakautuneen satunnaisluvun väliltä $[0, 1]$ ja valitsemalla sen arvon, jonka kohdalle satunnaisluku osuu multinomijakauman kertymäfunktiossa.

Kun kaikille isättömille solmuille on saatu arvot, voidaan edetä solmuihin, jotka riippuvat yhdestä solmusta ja arpoa niille arvo kiinnittämällä isäsolmu aikaisemmin saatuun arvoonsa. Näin etenemällä saadaan kaikkia verkon solmuja vastaaville muuttujille arvo, jotka voidaan koota vektoriksi. Näytteitä voidaan arpoa lisää unohtamalla edelliset arvot ja aloittamalla prosessi alusta. Generoitujen näytteiden asymptoottiset suhteelliset lukumäärät vastaavat verkon todennäköisyysjakaumaa.

Luku 4

Tilanneriippuva graafinen malli

Kuten luvussa 2.2 kerrottiin, solussa aktivoituu eri tilanteissa erilaisia säätely-vuorovaikutuksia. Tässä luvussa esitellään menetelmä, joka pystyy löytämään eroavaisuuksia säätelyssä eri olosuhteissa suoritettujen ilmentymismittausten välillä. Menetelmä perustuu Bayes-verkkomalliin, joka pystyy esittämään tilannekohtaisia riippuuvuuksien eroja. Ne osat riippuvuusrakenteesta, joissa ei ole eroja luokkien välillä, estimoidaan kaikkiin luokkiin kuuluvien näytteiden perusteella, mikä antaa luotettavan tuloksen.

Luvussa 4.2 kerrotaan miten yhdellä Bayes-verkolla voidaan käsitellä tilannekohtaisia eroja säätelyssä. Luvussa 4.3 on esitelty opetusalgoritmi, joka pystyy automaattisesti tunnistamaan verkon rakenteessa ne kohdat, joissa on eroja luokkien välillä. Opetuksen vaatimasta ajasta on kerrottu luvussa 4.5.

4.1 Luokkamuuttujan käsittelytapoja

Jos halutaan, että verkko pystyy käsittelemään vuorovaikutuksia, jotka toimivat vain joissain olosuhteissa, täytyy verkkoon jollain tavalla sisällyttää tieto näyteluokasta eli olosuhteesta, jossa mittaus on suoritettu. Suoraviivaisin tapa on lisätä verkkoon uusi muuttuja, jonka arvoina on mahdolliset luokat. Kaikki solmut, jotka käyttäytyvät eri tavalla eri luokissa, voidaan laittaa luokkasolmun lapsiksi, jolloin niiden jakauma voi riippua luokasta.

Luokkamuuttujan mukaan ottamiseen on kaksi, toisistaan periaatteiltaan poikkeavaa tapaa. Generatiivisessa opetuksessa luokkamuuttuja c on vain yksi lisämuuttujana yhteisjakaumassa $p(c, x)$. Toinen vaihtoehto on diskrimatiivinen mallinnus, missä tavoitteena on oppia mahdollisimman hyvä ennustin $p(c|x)$ luokkamuuttujan arvolle kun muiden muuttujien arvot on havaittu.

Tässä luvussa esiteltävä menetelmä perustuu generatiiviseen lähestymistapaan. Jos havainnot ovat peräisin malliperheeseen kuuluvasta mallista, pystyy generatiivinen opetus identifioimaan oikean mallin kunhan näytteitä on käytettävis-

sä tarpeeksi. Kaikkien Bayes-verkkoja käyttävien geenien säätelyä kuvaavien mallien perusoletuksena on, että ilmentymisprofiilit ovat peräisin Bayes-verkon kaltaisesta todennäköisyysjakaumasta, eli että opitun verkon rakennetta tulkitsemalla pystytään päättelemään todellisia säätelysuhteita.

Diskriminatiivisen opetuksen tavoitteena on muodostaa luokkamuuttujaa ennustava verkko. Tämän saavuttamiseksi verkon täytyy kuvata luokkien eroja, mikä vaikuttaisi ensialkuun hyvin tärkeältä ominaisuudelta olosuhderiippuvaisten geenisäätelyiden etsimisessä, mutta diskriminatiivinen opetus ei takaa, että löydetty verkko vastaisi todellisia eroja säätelyssä. Yksinkertainen esimerkki on tilanne, jossa eräs geeni on aina tilassa A ensimmäisessä luokassa ja tilassa B toisessa luokassa. Tämä geeni riittää yksinään ennustamaan luokan täydellisesti. Siksi riittää kytkeä sitä vastaava solmu luokkasolmuun ja jättää muut solmut kokonaan yhdistämättä toisiinsa. Selvästikään $p(c|x)$ ei kerro mitään geenien välisistä riippuvuuksista.

4.2 Tilanneriippuva verkkorakenne

Jos olisi saatavilla hyvin paljon ilmentymismittauksia eri tilanteista, optimaalisin ratkaisu olisi jakaa näytteet ryhmiin luokittain ja opettaa joka luokalle oma Bayes-verkko. Tällä tavalla saataisiin selville erikseen kunkin luokan optimaalisin riippuvuusrakenne. Valitettavasti silloin jokaisen luokan opettamiseen voitaisiin käyttää vain kyseisen luokan näytteitä, mikä entisestään pienentäisi näytemäärää ja johtaisi todennäköisesti ylisovittumiseen. Riskiä voi tietenkin vähentää rajoittamalla verkon rakennetta siten, että efektiivinen parametrien määrä pienenee. Joitain esimerkkejä malliperheen rajaamisesta on esitelty luvussa 3.3. Seuraavaksi esitellään uusi malli, jossa luokkien säätelyerot voidaan esittää yhdellä Bayes-verkolla. Erona aikaisempiin menetelmiin on se, miten luokkamuuttujaa käsittellään erikoistapauksena solmujen paikallisissa jakaumissa.

Yleensä voidaan olettaa, että vain pieni osa säätelyvuorovaikutuksista muuttuu eri olosuhteissa ja suurin osa pysyy samoina. Kannattaa käyttää muuttumattomien riippuvuuksien estimoimiseen kaikkien luokkien havaintoja ja estimoida vain luokkariippuvat osat luokkakohtaisten havaintojen perusteella. Silloin luokasta riippumattomien verkon osien estimointi on mahdollisimman luotettavaa, koska käytettävissä on paljon opetusnäytteitä. Ongelmaksi jää tunnistaa ne riippuvuudet, jotka ovat samoja eri luokissa. Tämän ongelman ratkaisemiseksi esitellään seuraavassa aliluvussa opetusalgoritmi, joka automaattisesti löytää luokkien riippuvuuserot, mutta aluksi perehdytään siihen, miten tilanneriippuvuudet voidaan esittää verkon rakenteessa.

Tilanneriippuvilla säätelyvuorovaikutuksilla tarkoitetaan tässä vain jossain luokassa esiintyviä riippuvuuksia. Ne voidaan esittää säännöllisyyksinä solmun paikallisessa jakaumassa. Jos esimerkiksi solmun X jakauma riippuu isäsolmu-

jen $\boldsymbol{U}_1$ arvoista luokassa i ja isäsolmuista $\boldsymbol{U}_1$ ja $\boldsymbol{U}_2$ luokassa $j \neq i$, eli jos

$$\begin{cases} p(X|C = i, \boldsymbol{U}_1, \boldsymbol{U}_2) = p_1(X|\boldsymbol{U}_1) \\ p(X|C = j, \boldsymbol{U}_1, \boldsymbol{U}_2) = p_{12}(X|\boldsymbol{U}_1, \boldsymbol{U}_2), \end{cases} \tag{8}$$

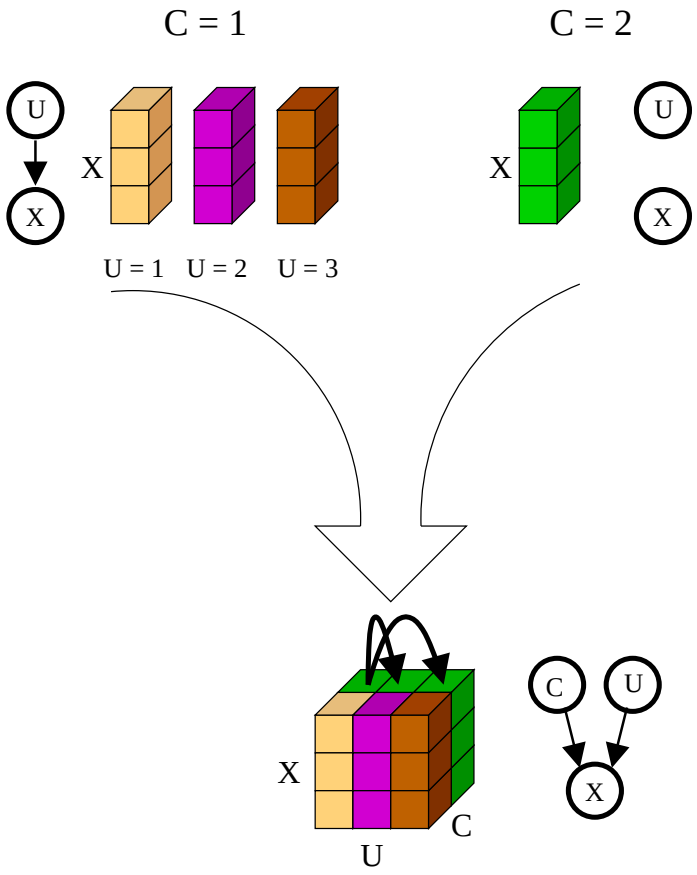
niin solmun X jakauman määrittelemiseksi täytyy kertoa multinomijakauman parametrit jokaista isäsolmujen $\boldsymbol{U}_1$ konfiguraatiota kohden luokassa i ja jokaista isäsolmujen $\boldsymbol{U}_1$ ja $\boldsymbol{U}_2$ konfiguraatiota kohden luokassa j. Toisella tavalla ilmaistuna luokassa i käytetään samoja parametrejä sellaisissa isäsolmukonfiguraatioissa, joiden ainot erot ovat solmujen $\boldsymbol{U}_2$ arvoissa. Tätä on havainnollistettu kuvassa 9. Ehdossa (8) jompikumpi säätelijäjoukoista voi olla myös tyhjä. Jos esimerkiksi $\boldsymbol{U}_2$ on tyhjä, ovat solmun X jakaumat luokissa i ja j erilaiset vaikka isäsolmut ovatkin samat. Käytännössä se tarkoittaa, että moduulin eri luokissa erilainen käyttäytyminen on seurausta jostain piilomuuttujasta, jota mallissa ei ole mukana.

Luokkariippuvien isäsolmujen esittämiseksi pitää verkkoon ottaa mukaan luokkasolmu. Solmut, joiden jakauma riippuu luokasta esimerkiksi koska solmulla on luokissa erilaiset riippuvuudet, ovat sen lapsia. Tätä verkkoa nimitetään *perusverkoksi*. Solmun i isäsolmuja (ilman luokkamuuttujaa) luokassa c kutsutaan *luokkakohtaisiksi säätelijöiksi* ja niille käytetään merkintää $\boldsymbol{U}_i^c$. Luokkakohtaisten säätelijöiden ei tarvitse olla toisensa poissulkevia vaan sama isäsolmu voi esiintyä useammassa luokassa. Niiden yhdiste

$$\boldsymbol{U}_i' = \bigcup_{c \in \mathrm{Val}(C)} \boldsymbol{U}_i^c$$

on nimeltään *varsinaisten säätelijöiden* joukko, koska se sisältää kaikki jossain luokassa solmuun suoraan vaikuttavat säätelijät mutta ei luokkamuuttujaa. Jos solmulla on eri säätelijät eri luokissa, niin perusverkossa sen isäsolmuina on varsinaiset säätelijät ja luokkamuuttuja, koska niitä kaikkia tarvitaan jakauman esittämiseen. Jos solmun jakauma ei riipu luokasta, sen isäsolmuina perusverkossa on pelkät säätelijät ilman luokkamuuttujaa. Perusverkon rakenteesta ei suoraan käy ilmi mitkä säätelijät liittyvät mihinkin luokkaan. Sitä varten pitää tutkia solmujen jakaumien säännöllisyyksiä. Siksi määritellään joka solmulle jakauman säännöllisyydet $\mathcal{L} = \{L_1, L_2, \ldots, L_M\}$, missä kuvaus $L_i(c) \subseteq \{1, 2, \ldots, \boldsymbol{U}_i'\}$ kertoo niiden solmujen i varsinaisten säätelijöiden indeksit (jossain ennalta sovitussa järjestyksessä), joista solmun jakauma luokassa c riippuu.

Ehto (8) ja sen esittämiseen käytettävät säännöllisyyskuvaukset $\mathcal{L}$ ovat versioita luvussa 3.2.3 esitellystä kontekstuaalisesta riippumattomuudesta. Aikaisemmissa tutkimuksissa konteksti, joka kertoo mistä muuttujista tutkittavan solmun arvo riippuu, määräytyi päätöspuun tai päätösverkon perusteella. Tässä työssä käytetyt jakaumien säännöllisyydet $\mathcal{L}$ rajoittavat kontekstin pelkäksi luokkamuuttujaksi. Jos $L_i(c)$ on jossain luokassa $c \in \mathrm{Val}(C)$ aito osajoukko varsinaisista säätelijöistä $\boldsymbol{U}_i'$, niin kyseinen luokkamuuttujan arvo määrää solmun jakauman kontekstuaalisesti riippumattomaksi muista varsinaisista säätelijöistä $\boldsymbol{U}_i' \setminus L_i(c)$. Rajoitus pelkkään luokkamuuttujaan on tehty tulkittavuuden vuoksi; jakaumien säännöllisyydet L kertovat suoraan missä luokissa



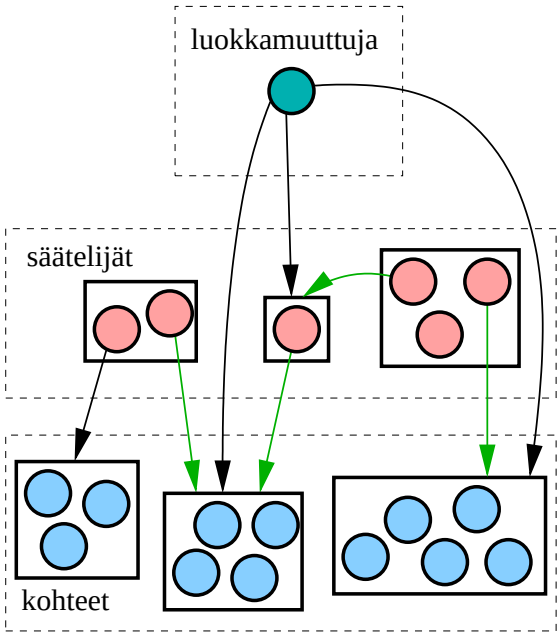
Kuva 9: Tilanneriippuvassa graafisessa mallissa perusverkon paikallisten jakaumien parametrit kootaan luokkakohtaisista parametreistä (kuvissa esitetty vain solmun X parametrit). Ensimmäisessa luokassa (ylärivi vasemmalla) U on X:n isäsolmu, joten jokaista U:n mahdollista arvoa, joita tässä on kolme kappaletta, kohden on olemassa erillinen multinomijakauma X:lle. Jakaumat on esitetty pylväinä, joissa jokainen pieni kuutio sisältää todennäköisyyden havaita X yhdessä kolmesta mahdollisesta tilasta. Eri värit liittyvät eri isäsolmun arvoihin. Toisessa luokassa (ylärivi oikealla) U ei ole X:n isäsolmu, eli X noudattaa samaa jakaumaa U:n arvoista riippumatta. Alarivillä luokkakohtaiset jakaumat on yhdistetty. Koska luokissa on eri isäsolmut, täytyy tilanneriippuvassa verkossa X:n isäsolmuna olla luokkamuuttuja. Luokkakohtaiset isäsolmut eivät käy ilmi alarivin verkkorakenteesta vaan parametritaulukosta, jossa toisessa luokassa käytetään samoja parametrejä kaikilla U:n arvoilla.

isäsolmut vaikuttavat. Aikaisemmissa menetelmissä saman tiedon esiin kaivaminen on paljon työläämpää.

Yksi perusverkko sisältää itseasiassa useita verkkoja, yhden jokaista luokkaa kohden. Ne voidaan erotella erillisiksi verkoiksi kiinnittämällä luokkamuuttuja vuorollaan jokainen mahdolliseen arvoonsa ja poistamalla perusverkosta kaaret, jotka eivät esiinny paikallisten jakaumien riippuvuuksissa käsiteltävässä luokassa. Kun vielä lopuksi syntyneistä verkoista poistetaan luokkasolmu ja kaikki siitä lähtevät kaaret, on tuloksena luokkakohtaiset verkot.

Luokkakohtaisten säätelijöiden lisäksi toinen esiteltävän mallin erityisominai-

suus on moduuliverkkoarkitehtuurin käyttö perusverkkona. Moduuliverkot esiteltiin luvussa 3.3.1. Koska geenit toimivat luonnostaan ryhminä, on moduulien käyttö perusteltua. Ne myös vähentävät tehtävän monimutkaisuutta ja helpottavat verkon oppimista. Verkossa on kahdenlaisia moduuleita: *säätelijämoduulit* koostuvat pelkästään potentiaalisista säätelytekijöistä ja loput geenit sijoitellaan *kohdemoduuleihin*. Potentiaaliset säätelijät ovat geenejä, joiden tiedetään kirjallisuuden perusteella tuottavan säätelytekijöitä (katso luku 2.5). Ne valitaan ennen verkon muodostamista. Jotta opetus keskittyisi mallintamaan nimenomaan säätelyä, sallitaan säätelijä- ja kohdemoduulien isäsolmut (mahdollisen luokkamuuttujan lisäksi) pelkästään potentiaalisten säätelijöiden joukosta. Tällöin mallin täytyy yrittää selittää mahdolliset erot säätelyvuorovaikutusten avulla. Jokaisella moduulilla saa olla korkeintaan d (tyypillisesti noin 3–5) isäsolmua, mikä on biologisesti perusteltua, koska tunnetusti suurinta osaa geeneistä säätelee vain muutama säätelytekijä [35] ja toisaalta monimutkaisempia riippuvuuksia ei kuitenkaan pystyttäisi oppimaan luotettavasti. Rajoitus auttaa myös verkon opetuksessa sillä se vähentää kokeiltavien verkkorakenteiden määrää. Kuvassa 10 on pieni esimerkkiverkko.



Kuva 10: Perusverkon rakenne. Vain kaaret säätelijäsolmuista kohdesolmuihin ja luokkasolmuista säätelijäsolmuihin tai sellaisiin kohdesolmuihin, joiden isäsolmuna on vähintään yksi säätelijä, ovat sallittuja. Kaaret, joiden olemassaolo voi riippua luokkamuuttujan arvosta, on värjätty vihreällä. Kuvasta ei näy mihin luokkaan kaaret liittyvät; sitä varten pitää tarkastella moduulien paikallisten jakaumien säännöllisyyksiä.

Tilanneriippuvassa mallissa oletetaan, että geenin eri tilanteissa erilainen käyttäytyminen johtuu säätelyn eroista. Opittavan mallin ohjaamiseksi tähän suuntaan asetetaan kohdemoduuleille vielä yksi lisärajoitus; kohdemoduulin isäsolmuna saa olla luokkasolmu vain, jos sillä lisäksi on vähintään yksi säätelijäisäsolmu. Rajoitus pakottaa mallin yrittämään selittää mahdollisia luokkakohtaisia eroja geenien ilmentymisprofiileissa tilanneriippuvasta säätelystä johtu-

viksi, mikä auttaa opitun mallin tulkitsemisessa. Säätelijämoduuleilla tällaista rajoitusta ei ole ja niiden käyttäytymistä opetusalgoritmi voi joskus päätyä selittämään riippuvan pelkästään luokkamuuttujasta. Tällaisen käyttäytymisen voi tulkita johtuvan jostain piilomuuttujasta, joka ei ole mukana mallissa, koska muuten opetusalgoritmi todennäköisesti lisäisi sen isäsolmuksi luokkamuuttujan sijaan. Säätelytekijöiden kohdalla tämä onkin mahdollista, koska ne voivat riippua solun ulkopuolelta tulevista signaaleista.

Tilanneriippuvan verkon määrittelemiseksi täytyy siis kertoa perusverkon rakenteen $\mathcal{S}$, solmujen sijoittelun $\mathcal{A}$ ja paikallisten jakaumien parametrien $\boldsymbol{\theta}$ lisäksi moduulien paikallisten jakaumien säännöllisyydet $\mathcal{L} = \{L_m\}$, jotka määräävät mitkä perusverkon isäsolmuista ovat aktiivisia kussakin luokassa.

Tapa esittää luokkakohtaiset säätelijät jakaumien säännöllisyyksinä ja käytetty versio moduuliverkoista eivät ole sidoksissa toisiinsa. Niitä voitaisiin aivan hyvin käyttää myös toisistaan riippumatta.

4.3 Opetusalgoritmi

Edellisessä luvussa esitellyn mallin opetus onnistuu melkein samalla tavalla kuin moduuliverkkojen opetus (ks. luku 3.3.1). Perusajatus on samanlainen ahne haku, joka vuorottelee rakennehaun ja solmujen sijoittelun optimoinnin välillä kunnes kumpikaan vaihe ei tuota verkkoon muutoksia. Luokkamuuttujan mukaan ottamisesta ja hieman yleistä moduuliverkkoa tiukemmista rakenteen rajoituksista seuraa opetukseen muutamia muutoksia, joita on käsitelty tarkemmin alla.

4.3.1 Tilanneriippuvan graafisen mallin yhteensopivuusmitta

Opetusnäytteen luokkaa käsitellään ylimääräisenä muuttujana. Opetusaineisto on siis joukko vektoreita $D = \{(c^{(1)}, \boldsymbol{x}^{(1)}), \ldots, (c^{(K)}, \boldsymbol{x}^{(K)})\}$, missä $c^{(i)}$ ja $\boldsymbol{x}^{(i)}$ ovat i:nnen näytteen luokka ja ekspressioprofiili. Ennen opetuksen käynnistämistä on alustettava solmujen sijoittelu. Tämä suoritetaan klusteroimalla keskenään samankaltaiset muuttujat[5] ryhmiin k:n keskiarvon klusterointimenetelmällä. Siinä muuttujat jaetaan aluksi moduulien lukumäärän mukaiseen määrään satunnaisia klustereita, joita sen jälkeen päivitetään siirtämällä muuttujia yksi kerrallaan klusterista toiseen siten, että klusterien sisäinen hajonta pienenee. Satunnaisalustuksen vaikutuksen pienentämiseksi klusterointi toistetaan kymmenen kertaa lähtien eri alkuarvoista ja lopputulokseksi valitaan paras klusterointi. Koska säätelijä- ja kohdesolmujen täytyy olla omissa moduuleissaan, klusterointi suoritetaan erikseen molemmille solmutyypeille.

[5] Yleensä klusteroinnissa pyritään ryhmittelemään samanlaiset *havainnot*, mutta nyt halutaan alkuarvaus muuttujien sijoittelulle moduuleihin, siksi ryhmitellään *muuttujat*.

Yhteensopivuusmittana käytetään moduuliverkon BD-funktiota (6), jota muutetaan huomioimaan paikallisten jakaumien mahdollinen luokkariippuvuus. Seuraava lauseke perustuu Friedmanin ja Goldszmidtin [14] esitykseen BD-mitan laajentamisesta multinomijakaumille, joissa on säännöllisyyksiä. Bayesiläiseksi yhteensopivuusmitaksi voidaan johtaa samalla tavalla kuin normaalien Bayesverkkojen yhteydessä (vertaa lausekkeeseen (2))

$$score(\mathcal{A}, \mathcal{S}, \mathcal{L}|D) = \log p(D|\mathcal{A}, \mathcal{S}, \mathcal{L}) + \log p(\mathcal{L}|\mathcal{A}, \mathcal{S}) + \log p(\mathcal{A}, \mathcal{S}).$$

Summan ensimmäinen termi voidaan laskea samalla tavalla kuin moduuliverkossa (5), kunhan huomioidaan, että jakaumien säännöllisyydet $\mathcal{L}$ kertovat mitkä isäsolmukonfiguraatiot liittyvät samoihin tyhjentäviin tunnuslukuihin. Samoin viimeinen termi, eli perusverkon rakennepriori, lasketaan kuten moduuliverkoissa. Keskimmäinen termi on paikallisen jakauman sännöllisyyksien prioritodennäköisyys. Friedman ja Goldszmidt ehdottavat prioriksi kuvauspituutta.

Moduulin m jakauman säännöllisyyksien L_m kuvauspituus on MDL-periaatteen mukaisesti informaation määrä, joka tarvitaan moduulin eri luokkien säätelijöiden esittämiseen. Aluksi tarvitaan yksi bitti kertomaan riippuvatko isäsolmut luokasta vai vastaako jokaista isäsolmukonfiguraatiota oma jakauma. Jos moduuli ei ole luokkamuuttujan lapsi, niin muuta ei tarvitsekaan tietää, koska silloin ainoa vaihtoehto on oma jakauma jokaiselle isäsolmukonfiguraatiolle. Jos moduulin jakauma riippuu luokasta, pitää joka luokalle kertoa erikseen isäsolmut. Mahdolliset isäsolmut ovat osajoukko moduulin varsinaisista säätelijöistä $\boldsymbol{U}'_m$. Luokkakohtaiset isäsolmut voidaan esittää kertomalla aluksi varsinaisten säätelijöiden lukumäärä, mihin kuluu $\log_2(|\boldsymbol{U}'_m|)$ bittiä. Lisäksi pitää jokaisen luokan kohdalla kertoa monesko (jossain etukäteen sovitussa järjestyksessä) luokan isäsolmujoukko on kaikista $\binom{|\boldsymbol{U}'_m|}{|\boldsymbol{U}^c_m|}$ saman kokoisesta joukosta, mikä vaatii $\log_2 \binom{|\boldsymbol{U}'_m|}{|\boldsymbol{U}^c_m|}$ bittiä. Yhden moduulin paikallisen jakauman säännöllisyyksien kuvauspituus on siis

$$KP(L_m|\boldsymbol{U}_m) = \begin{cases} 1 + \log_2(|\boldsymbol{U}'_m|) + \sum_{c \in C} \log_2 \binom{|\boldsymbol{U}'_m|}{|\boldsymbol{U}^c_m|}, & \text{kun } C \in \boldsymbol{U}_m \\ 1, & \text{kun } C \notin \boldsymbol{U}_m \end{cases}$$

Koska kuvauspituus ei riipu solmujen sijoittelusta $\mathcal{A}$, sijoittelu ei vaikuta myöskään jakaumien säännöllisyyksien prioriin $p(\mathcal{L}|\mathcal{A}, \mathcal{S}) = p(\mathcal{L}|\mathcal{S})$. Priori on verrannollinen kaikkien moduulien kuvauspituuksien summaan:

$$p(\mathcal{L}|\mathcal{S}) \propto \left(2^{-\sum_m KP(L_m|\boldsymbol{U}_m)}\right).$$

Kirjoittamalla lauseke auki osoittautuu, että priorin logaritmi voidaan lausua summana moduulikohtaisista termeistä:

$$\log p(\mathcal{L}|\mathcal{S}) \propto \sum_{m=1}^{M} \eta_{m,c}(\boldsymbol{U}_m),$$

missä priori jakautuu osiin moduuleittain:

$$\eta_{m,c}(\boldsymbol{U}_m) = \begin{cases} -\log(2) - \log(|\boldsymbol{U}'_m|) - \sum_c \log \binom{|\boldsymbol{U}'_m|}{|\boldsymbol{U}^c_m|}, & \text{kun } C \in \boldsymbol{U}_m \\ -\log(2), & \text{kun } C \notin \boldsymbol{U}_m \end{cases}$$

Termi $-\log(2)$ ja priorin normeerausvakio (jonka arvoa ei ole tässä edes laskettu) voidaan jättää huomiotta parasta rakennetta etsittäessä, koska ne ovat samoja kaikille rakenteille.

Koko yhteensopivuusmitta on summa moduulien paikallisista mitoista samalla tavalla kuin moduuliverkossa kaavassa (6), mutta nyt paikallisissa yhteensopivuusmitoissa on lisäksi jakauman kuvauspituuteen liittyvä termi $\eta_{m,c}(\boldsymbol{U}_m)$:

$$score_m^M(A_m, \boldsymbol{U}_m, L_m, D) = \sum_{c \in \mathrm{Val}(C_m)} \left[\sum_{j \in \mathrm{Val}(\boldsymbol{U}_m^c)} \Phi^M(m, \nu(c,j)) + \eta_{m,c}(\boldsymbol{U}_m) \right] + \rho_m(\boldsymbol{U}_m) + \kappa_m(A_m), \quad (9)$$

missä $\mathrm{Val}(C_m)$ on luokkamuuttujan arvojoukko $\mathrm{Val}(C)$ niille moduuleille, joiden jakauma riippuu luokasta, ja tyhjä joukko muille. Moduulikohtaiset uskottavuusfunktion $\Phi^M(m, \nu(c,j))$, verkkorakenteen priorin $\rho_m(\boldsymbol{U}_m)$ ja solmujen sijoittelun priorin $\kappa_m(A_m)$ termit ovat samoja kuin moduuliverkoissa. Funktio $\nu(c,j)$ kertoo mitkä perusverkon isäsolmut ovat kussakin luokassa aktiivisia. Jos isäsolmukonfiguraatiot j ja j' eroavat ainoastaan sellaisten isäsolmujen osalta, joista moduulin jakauma ei riipu luokassa c, niin $\nu(c,j) = \nu(c,j')$. Jos moduulin jakauma ei riipu luokasta paikallisen yhteensopivuusmitan lauseke yksinkertaistuu moduuliverkon paikalliseksi mitaksi (7).

Lausekkeessa (9) rakenne- ja sijoitteluprioreista riippumaton osa jakautuu summaksi luokkakohtaisia termejä. Opetusalgoritmin kannalta tämä tarkoittaa, että luokkakohtaiset isäsolmut voidaan etsiä muista luokista riippumatta samalla tavalla kuin mitan jakautuminen osiin moduuleittain sallii moduulien optimoinnin yksi kerrallaan. Seuraavaksi esiteltävä opetusmenetelmä käyttää tätä hyväksi.

Joissain tilanteissa haluttaisiin ehkä käyttää monipuolisempaa säännöllisyysprioria $p(\mathcal{L}|\mathcal{S})$. Todennäköisyys valita isäsolmu jollekin luokalle voisi esimerkiksi olla suurempi, jos sama isäsolmu jo esiintyy jossain toisessa luokassa. Tällainen priori olisi kyllä vapaampi, mutta kaikkien luokkien luokkakohtaiset säätelijät pitäisi valita yhdellä kertaa. Koska kokeiltavia isäsolmuyhdistelmiä olisi silloin paljon enemmän kuin esiteltyä prioria käytettäessä, jolloin isäsolmut voidaan valita erikseen joka luokalle, haku olisi paljon hitaampaa.

Tilanneriippuvalle graafiselle mallille voidaan johtaa BD-yhteensopivuusmitta melko suoraviivaisesti moduuliverkon yhteensopivuusmitan perusteella. Solmujen paikallisten jakaumien mahdolliset luokkariippuvuudet huomioidaan käyttämällä samoja parametrejä useille isäsolmukonfiguraatioille. Säännöllisyyksille voidaan johtaa priori minimikuvauspituusperiaatteen pohjalta.

4.3.2 Sijoittelun optimointi

Solmujen sijoittelun optimointi (algoritmi 2) tapahtuu suurimmaksi osaksi sa-

Algoritmi 2 Sijoittelun optimointi

Syöte: Solmujen sijoittelu $\mathcal{A}$, verkon rakenne $\mathcal{S}$ ja opetusnäytteet D
Tuloste: Päivitetty sijoittelu

```
repeat
  for n ∈ solmut do
    for m ∈ sallitut moduulit do
      Â ← A
      Â[n] ← m
      if (score(Â, S|D) > score(A, S|D)) ja syklitön(Â, S) then
        A ← Â
      end if
    end for
  end for
until ei muutoksia sijoitteluun
Tulosta A
```

malla tavalla kuin moduuliverkossa. Moduulien jakaumien luokkariippuvuudet eivät vaikuta siihen, koska solmujen sijoittelu ei riipu luokasta ja säätelijöiden luokkariippuvuus näkyy vain moduulien isäsolmuissa, joita sijoittelun haussa ei muuteta. Toinen muutos, eli moduulien jako säätelijä- ja kohdemoduuleihin, sen sijaan tuo muutoksia myös sijoittelun hakuun. Solmuja saa nimittäin siirtää vain oikean tyyppisten moduulien välillä, potentiaalisia säätelytekijöitä vain säätelijämoduulien välillä ja muita solmuja vain kohdemoduulista toiseen. Päätös solmun siirtämisestä uuteen moduuliin tehdään pelkästään siirron lähde- ja kohdemoduulien paikallisissa yhteensopivuusmitoissa aiheuttamien muutoksien perusteella. Paikalliset mitat (9) taas riippuvat moduulin solmujen ja isäsolmujen arvoista. Sijoittelun optimointi kannattaa siis hoitaa kahdessa vaiheessa: aluksi säätelijöille etsitään paras sijoittelu kokeilemalla siirtää niitä vuorotellen jokaiseen toiseen säätelijämoduuliin kunnes yhdellekään säätelijälle ei löydy parempaa sijoitusta. Kohdesolmujen sijoittelu ei vaikuta säätelijöiden sijoittelun optimointiin, koska kohdesolmut eivät voi olla samassa moduulissa säätelijöiden kanssa tai niiden isäsolmuina. Toisessa vaiheessa optimoidaan kohdesolmujen sijoittelu pitäen säätelijät paikoillaan niille jo löydetyissä parhaissa mahdollisissa moduuleissa. Kohdesolmujen siirtely ei enää vaikuta säätelijöiden sijoitteluun, eli kun yhdenkään kohdesolmun siirtäminen ei enää kasvata yhteensopivuusmittaa, voidaan koko sijoittelun optimointi -vaihe lopettaa. Luokkasolmu pidetään koko ajan yksinään omassa moduulissaan.

Ennen siirron hyväksymistä täytyy tarkistaa, ettei siirto tee verkosta syklistä. Syklisyyden testaus tehdään aina perusverkossa, eli syklisyyden testauksessa moduulin isäsolmuiksi luetaan kaikkien luokkien isäsolmut. Syklisyyden testauksesta voi tehdä kolme havaintoa; ensinnäkin, kuten moduuliverkkojen yhteydessä luvussa 3.3.1 todettiin, syklisyyttä tarvitsee tarkastella vain moduulien muodostaman verkon, ei yksittäisten solmujen, tasolla. Toiseksi riittää testata säätelijämoduulien muodostaman osaverkon syklisyys. Tämä on riittävä ehto, koska kohdesolmut eivät voi olla minkään solmun isäsolmuja ja toisaalta luokkasolmulla ei saa olla isäsolmuja. Kohde- ja luokkasolmut eivät siis koskaan

Algoritmi 3 Rakennehaku

Syöte: Solmujen sijoittelu $\mathcal{A}$, verkon rakenne $\mathcal{S}$ ja opetusnäytteet D
Tuloste: Päivitetty rakenne

repeat
  for $m \in$ moduulit **do**
    for $n \in \{C \cup$ säätelijäsolmut$\}$ **do**
      $\hat{\mathcal{S}} \leftarrow \mathcal{S}$
      Lisää tai poista kaari solmusta n moduuliin m rakenteessa $\hat{\mathcal{S}}$
      if $C \in \boldsymbol{U}_m$ **then**
        Valitse parhaat säätelijäosajoukot moduulin m luokille $\hat{S}$:ssä
      end if
      if $(score(\mathcal{A}, \hat{\mathcal{S}}|D) > score(\mathcal{A}, \mathcal{S})|D)$ ja syklitön$(\mathcal{A}, \hat{\mathcal{S}})$ **then**
        $\mathcal{S} \leftarrow \hat{\mathcal{S}}$
      end if
    end for
  end for
until ei muutoksia rakenteeseen
Tulosta $\mathcal{S}$

voi olla osa sykliä. Syklisyystarkistusta tarvitaan siis vain säätelijäsolmujen sijoittelua haettaessa, ei kohdesolmujen sijoittelun optimoinnissa. Kolmanneksi selvästikin syklittömästä verkosta voi tulla syklinen vain lisättäessä kaaria, ei niitä poistettaessa. Sijoittelun optimoinnin kannalta tämä kolmas huomio tarkoittaa, että syklisyyden tarkistaminen on tarpeen vain siirrettäessä sellaista solmua, joka on jonkin säätelijämoduulin isäsolmu, koska tällaisen solmun siirtäminen lisää kaaren uuden moduulin ja siirrettävän solmun lapsisolmujen välille.

4.3.3 Rakennehaku

Rakennehaku (algoritmi 3) vaatii hieman sijoittelun optimointia suurempia muutoksia tavalliseen moduuliverkon rakenteenhakuun verrattuna. Perusidea on sama: kaikille yhden kaaren muutoksille lasketaan yhteensopivuusmitan muutos ja niistä paras hyväksytään. Iteraatiota jatketaan kunnes yksikään muutos ei enää kasvata mittaa. Käytetyn mittafunktion jakautuminen osiin moduuleittain tarkoittaa, että kun jotain verkon moduuleista muutetaan, tarvitsee laskea vain siihen moduuliin liittyvän osan muutos. Muiden moduulien paikalliset mitat pysyvät samoina.

Rakenteeseen tehtävien muutoksissa luokkamuuttuja on erityisasemassa, koska sen arvo määrää mistä varsinaisten säätelijäsolmujen osajoukosta moduulien arvot kyseisessä luokassa riippuvat. Niillä moduuleilla, joiden isäsolmuihin luokkamuuttuja ei kuulu, uuden säätelijäisäsolmun lisääminen tai vanhan poistaminen onnistuu täsmälleen samalla tavalla kuin moduuliverkossa laskemalla ehdotetun muutoksen vaikutus yhteensopivuusfunktioon.

Luokan lapsisolmujen päivittäminen rakennehaussa on hieman monimutkaisempaa kuin tavallisissa moduuliverkoissa. Kun tällaiselle moduulille kokeillaan lisätä uutta isäsolmua, niin joka luokalle täytyy etsiä uudelleen parhaat luokkakohtaiset säätelijät evaluoimalla mitan luokkakohtainen osa kaikilla varsinaisten säätelijöiden (mukaan lukien juuri lisättävä isäsolmu) osajoukoilla ja valitsemalla niistä parhaat joka luokalle. Koko yhteensopivuusmitan muutos lasketaan summana luokkakohtaisista muutoksista. Vastaavasti poistettaessa säätelijäisäsolmu moduulista joka luokalle etsitään parhaat luokkakohtaiset isäsolmut jäljelle jäävien säätelijöiden joukosta. Vastaavanlainen haku suoritetaan myös silloin kun lisätään luokkamuutuja uudeksi isäsolmuksi moduulille, joka ei aikaisemmin riippunut luokasta. Kun luokkamuuttuja poistetaan moduulin isäsolmujen joukosta, asetetaan moduulin uusiksi isäsolmuiksi kaikki varsinaiset säätelijäsolmut eli kaikki säätelijät, jotka ennen muutosta jossain luokassa säätelivät moduulia.

Jos luokasta riippuvalla moduulilla on kaaren lisäämisen tai poistamisen jälkeen luokkamuuttujan lisäksi $|\boldsymbol{U}'|$ isäsolmua, niin parhaiden luokkakohtaisten säätelijöiden löytämiseksi pitää laskea yhteensopivuusmitan arvo $2^{|\boldsymbol{U}'|}$ osajoukolle (mukaan lukien tyhjä joukko) joka luokassa. Isäsolmujen hakua voi nopeuttaa kahdessa erityistapauksessa; kun lisätään uusi säätelijä moduuliin, joka jo ennestään riippui luokkamuuttujasta, tarvitsee tutkia vain ne osajoukot, jotka sisältävät lisättävän isäsolmun, koska tiedetään, että paras muista osajoukoista oli valittuna ennen muutosta. Tällaisia osajoukkoja on $2^{|\boldsymbol{U}'|-1}$ kappaletta luokaa kohden. Toinen tapaus, jossa selvitään vähemmällä työllä, on säätelijän poistaminen. Jos poistettava säätelijä ei kuulu minkään luokan luokkakohtaisten isäsolmujen joukkoon, niin sen poistamisella ei voi olla vaikutusta moduuliin kyseisessä luokassa eikä näiden luokkien kohdalla moduuliin tarvitse tehdä mitään muutoksia. Täydellinen haku kaikista osajoukosta on tarpeen vain lisättäessä luokkamuuttuja isäsolmuksi moduuliin, jossa sitä ei aikaisemmin ollut, ja poistettaessa jokin luokkakohtaisista säätelijöistä. Pieni ajansäästö saavutetaan myös pitämällä koko ajan muistissa kulloisenkin verkon tyhjentävät tunnusluvut ja moduulien paikalliset yhteensopivuusmitan arvot. Silloin niitä ei tarvitse erikseen laskea aina sijoittelun tai rakenteen haun alussa.

Rakennehaussa syklisyyden tarkistus on tarpeen vain lisättäessä kaari säätelijäsolmusta säätelijämoduuliin, koska edellä todettiin, että vain säätelijämoduulien muodostama osaverkko voi olla syklinen ja vain kaaren lisääminen voi tehdä syklittömästä verkosta syklisen. Syklisyyden testaus on syytä tehdä vasta siinä vaiheessa, kun kaikkien muutoksien yhteensopivuusmitat on jo laskettu ja ollaan valitsemassa niistä parasta. Jos syklisyyttä testataan jo aikaisemmin, voidaan hylätä turhaan joitain kaarien lisäyksiä, jotka evaluointihetkellä tekevät verkosta syklisen, mutta jotka myöhemmin, muutaman verkkoon tehdyn muutoksen jälkeen, voivat olla sallittuja.

4.4 Mallin tulkinta

Tuloksena saatavasta verkosta on helppo löytää luokkariippuvat säätelyvuorovaikutukset, koska jokaisen luokkamuuttujan lapsimoduulin kohdalla kerrotaan suoraan mistä säätelijöistä moduulin jakauma riippuu missäkin luokassa. Saman asian selvittäminen tavallisella Bayes-verkosta, jossa jokaiseen isäsolmukonfiguraatioon liittyy erillinen jakauma muista konfiguraatioista riippumatta, on paljon vaikeampaa. Sellaisen verkon rakenteesta ei suoraan näe luokkakohtaisia säätelijöitä vaan ne pitää erikseen päätellä paikallisista jakaumista, mutta tarkoitukseen sopiva menetelmä ei ole itsestään selvä. Suoraviivaisin tapa olisi testata tilastollisesti toteuttavatko jotkin isäsolmujen osajoukot tilanneriippuvuusehdon (8). Tässä on ainakin kaksi ongelmaa. Opetuksessa käytettävä yhteensopivuusmitta integroi yli moduulien paikallisten jakaumien parametrien, eli parametrien arvoja ei varsinaisesti koskaan lasketa. Tämä on tietenkin hyvä asia, koska tulokset ovat luotettavampia. Jos jakaumia haluttaisiin käyttää testissä, täytyisi parametreille kuitenkin lopuksi laskea piste-estimaatti, mikä siis hukkaisi osan informaatiosta. Vaikeampi ongelma on sopivan testin valinta ja testien valtava määrä, kun kaikkien moduulien jokainen isäsolmuosajoukko pitäisi testata, mikä johtaisi moninkertaiselle tilastolliselle testaukselle tyypillisiin ongelmiin kuten p-arvojen inflaatioon. Mallia, joka antaa suoraan tuloksena luokkakohtaiset säätelijät on siis huomattavasti helpompi tulkita.

Kaikissa Bayes-verkon opetusalgoritmeissa on joitain ongelmia. Ahne haku ei pysty takaamaan tuloksen hyvyyttä suhteessa globaaliin optimiin. Silti sitä käytetään, koska se on nopea. Opetusalgoritmista riippumatta opitun verkon rakennetta tulkittaessa ongelmaksi muodostuu se, että Bayes-verkkojen oppimisalgoritmit yrittävät löytää havaintoihin parhaiten sopivan verkon, mutta ne eivät pysty takaamaan opitun verkon kausaalisuutta. Ilmentymissäätelyn kannalta tämä tarkoittaa, että jos todellisuudessa säätelijä X vaikuttaa geenin Y ilmentymiseen, niin Bayes-verkon opetus saattaa palauttaa verkon, jossa Y on X:n isäsolmu, koska näiden kahden muuttujan yhteisjakauma ei kerro kumpaan suuntaan kaaren pitäisi osoittaa. Osittain tästä syystä käytettävässä tilanneriippuvassa verkossa isäsolmuiksi sallitaan vain etukäteen potentiaalisiksi säätelijöiksi tiedetyt geenit, minkä pitäisi osaltaan ohjata kaarien suuntien valitsemista. Toinen tulkintaongelma on epäsuorat vaikutukset. Jos muuttuja X vaikuttaa todellisuudessa muuttujaan Y vain epäsuorasti jonkin kolmannen muuttujan Z kautta, voi vähäisestä näytemäärästä johtuen opetusalgoritmi silti palauttaa verkon, jossa X ja Y on kytketty suoraan kaarella toisiinsa. Tämä on erityisen todennäköistä, jos muuttuja Z on piilomuutuja eli jos sen arvoja ei olla mitattu.

Toinen tilanneriippuvan verkon opetusalgoritmin heikkous on se, ettei se pysty arvioimaan opitun verkon kaarien luotettavuutta. On mahdollista, että on olemassa suuri joukko hieman erilaisia verkkoja, joille kaikille BD-funktion arvo on suurinpiirtein yhtä suuri. Kaaret ja moduulit, joiden muuttamisella ei ole suurta vaikutusta verkon ja havaintojen yhteensopivuuteen, eivät ole kovin luotettavia, koska se, miten opetusalgoritmi sattui asettamaan ne saataa

riippua opetusnäytteistä. Kaarien luotettavuuden arvioimiseksi on ehdotettu useamman kuin yhden verkon oppimista esimerkiksi muodostamalla verkoista MCMC-näytteitä [16, 34] tai bootstrap-menetelmällä [15] ja kaarien esiintymisosuuksien laskemista kaikista opituista verkoista. Jos kahden solmun välillä on kaari melkein kaikissa verkoissa, niin kyseinen kaari on selvästi tärkeä mallille. Verkoista voisi laskea myös kuinka usein tietty kaari kuuluu vain toiseen tai molempiin luokkiin. Valitettavasti ei ole mitään takeita, että kaikissa verkoissa geenit olisivat samoissa moduuleissa. Tämä tekee tulosten tulkinnasta vaikeampaa, koska säätelijä-kohdemoduuli-parien asemesta tulkittavana olisi säätelijä-kohdegeeni-pareja, joita on huomattavasti enemmän.

Esitellystä mallista näkee suoraan moduulien luokkakohtaiset säätelijät, mikä helpottaa tulkintaa huomattavasti. Aikaisemmin esitellyissä menetelmiä käytettäessä luokkakohtaisten säätelijöiden selvittäminen on ollut vaikeaa. Tilanneriippuvalla mallilla on samat heikkoudet kuin useissa muissakin Bayes-verkkojen opetusmenetelmissä; ahne haku ei ole optimaalinen ja opitun verkon luotettavuuden arviointi on vaikeaa.

4.5 Opetusalgoritmin aikavaativuus

Seuraavaksi lasketaan arvio opetuksen vaatimalle ajalle. Oletetaan, että säätelijäsolmuja on N_s ja säätelijämoduuleita M_s kappaletta. Vastaavasti oletetaan, että N_k kohdesolmua on jaoteltu M_k moduuliin. Koska luokkasolmu on aina yksinään omassa moduulissaan, on verkossa yhteensä siis $N = N_s + N_k + 1$ solmua $M = M_s + M_k + 1$ moduulissa.

Solmujen sijoittelun optiminti tapahtuu kahdessa peräkkäisessä vaiheessa. Tarkastellaan aluksi säätelijäsolmujen sijoittelun hakua. Eniten aikaa kuluu yhteensopivuusmitan evaluoimiseen. Kuten edellä on todettu käyttämällä hyväksi mitan hajoamista moduuleittain saavutetaan huomattava ajan säästö, kun siirrettäessä solmu moduulista M_{vanha} moduuliin M_{uusi} tarvitsee laskea vain näihin moduuleihin liittyvät yhteensopivuusmitan osien muutokset muiden moduulien osuuksien säilyessä ennallaan. Olettaen, että kaikkien moduulien tyhjentävät tunnusluvut ja moduulikohtaiset mitat on laskettu etukäteen, tarvitsee solmun siirron aiheuttaman mitan muutoksen laskemiseksi vain vähentää solmun tyhjentävät tunnusluvut lähdemoduulin tunnusluvuista ja lisätä ne kohdemoduulin tunnuslukuihin. Yhteensopivuusmitan muutos selviää laskemalla sen uudet arvot lähde- ja kohdemoduuleissa ja vähentämällä niistä vanhat arvot, jotka ovat tallessa edelliseltä iteraatiokierrokselta.

Moduulin paikallisen yhteensopivuusmitan (9) arvon laskeminen vaatii aikaa lineaarisesti suhteessa moduulin tyhjentävien tunnuslukujen määrään, joka puolestaan riippuu moduulin isäsolmujen määrästä. Edellä todettiin, että yhden solmun siirto moduulista toiseen vaatii kaksi paikallisten yhteensopivuusmittojen evaluaatioita. Yhdellä sijoittelun päivitysiteraatiolla jokaista solmua yritetään siirtää vuorollaan jokaiseen muuhun moduuliin kunnes löytyy mittaa

kasvattava sijoittelu, eli enintään täytyy tutkia $N_s(M_s - 1)$ muutosta. Segal et al. [43] huomauttaa, että jos moduulit ovat keskenään suurinpiirtein saman kokoisia, eli jos jokaisessa on noin N_s/M_s solmua, niin tarvittavien sopivuusmitan evaluaatioiden määrä on lähes lineaarinen solmujen määrän suhteen, $2(M_s - 1)N_s/M_s = \mathcal{O}(N_s)$. Iteraatiota toistetaan, kunnes yhdenkään solmun sijoittelu ei enää muutu. Vastaavalla päättelyllä voidaan osoittaa, että yksi kohdesolmujen sijoittelun haku -iteraatio laskee yhteensopivuusmitan arvon $\mathcal{O}(N_k)$ kertaa.

Rakennehaussa tehdään muutoksia moduulien isäsolmujen joukkoon. Jokaisen moduulin kohdalla käydään läpi kaikki potentiaaliset isäsolmut ja lasketaan yhteensopivuusmitan muutos, kun solmu lisätään isäsolmuksi (jos se ei jo aikaisemmin ollut kyseisen moduulin isäsolmu) tai poistetaan isäsolmujen joukosta (jos se jo oli siellä). Yhden moduulin kohdalla tarvitaan siis $\mathcal{O}(N_s)$ tyhjentävien tunnuslukujen laskemista ja yhteensopivuusmitan evaluaatiota. Rakennehaun alussa on tämä laskettava kaikille moduuleille, minkä kustannus on $\mathcal{O}(MN_s)$. Koska seuraavilla kierroksilla rakennetta muutetaan aina vain yhden moduulin osalta ja yhteensopivuusmitta on summa yksittäisistä moduuleista riippuvista termeistä, seuraavat askeleet vaativat vain $\mathcal{O}(N_s)$ paikallisen mitan arvon laskemista.

Joka kerta, kun luokasta riippuvien moduulien isäsolmuihin tehdään muutoksia, täytyy jokaiselle luokalle etsiä uudelleen parhaat säätelijät. Tähän tarvitaan aikaa pahimmillaan $\mathcal{O}(|C|2^{|\boldsymbol{U}'|}N_s)$, missä $|\boldsymbol{U}'|$ on varsinaisten säätelijöiden lukumäärä kaaren lisäämisen tai poistamisen jälkeen. Koska algoritmissa yhden moduulin isäsolmujen lukumäärä on rajoitettu $|\boldsymbol{U}'| \leq g$, pysyy tarvittava laskenta-aika aisoissa vaikka riippuvuus näyttääkin eksponentiaaliselta.

Verkon syklisyyden testausta tarvitaan sekä sijoittelun että rakenteen optimoinnissa. Luvussa 3.2.2 esiteltiin algoritmi, joka testaa verkon syklisyyden ajassa $\mathcal{O}(|E| + |V|)$, missä $|E|$ on verkon kaarien määrä ja $|V|$ solmujen määrä. Koska tilanneriippuvan mallin syklisyyden toteamiseen riittää tutkia säätelijämoduulien muodostamaa verkkoa, onnistuu syklisyyden testaus ajassa $\mathcal{O}(|E_s| + |M_s|)$, missä $|E_s|$ on säätelijämoduulien välisten kaarien lukumäärä. Säätelijämoduulien välisen verkon ylläpidosta algoritmin suorituksen aikana syntyy hieman lisäkustannuksia, mutta käytännössä se voidaan alustaa sijoittelun tai rakenteen optimointinin alussa ja sitä tarvitsee päivittää vain tehtäessä muutoksia säätelijäsolmujen välille.

Rakennehaku ja sijoittelun optimointi laskevat siis yhteensopivuusmitan muutoksen yhden iteraation aikana yhteensä $\mathcal{O}(N_s) + \mathcal{O}(N_k) + \mathcal{O}(|C|2^{|\boldsymbol{U}'|}N_s) = \mathcal{O}(N_k + |C|2^{|\boldsymbol{U}'|}N_s)$ kertaa. Mitan muutoksen selvittämiseksi tarvitsee laskea vain päivitettyjen moduulien uudet tyhjentävät tunnusluvut ja vähentää niistä vanhat tunnusluvut, jotka voidaan pitää muistissa edelliseltä kierrokselta. Lisäksi joka iteraatiolla täytyy testata verkon syklisyys $\mathcal{O}(|E_s| + |M_s|)$ kertaa.

Luku 5

Simulaatiokokeet

Uuden mallin käyttökelpoisuus selviää lopullisesti vasta kokeiltaessa sitä käytännössä. Tässä luvussa testataan tilanneriippuvan mallin opetusta tunnetusta verkosta keinotekoisesti generoitujen näytteiden avulla, jolloin opittua verkkoa voidaan helposti verrata oikeaan verkkoon.

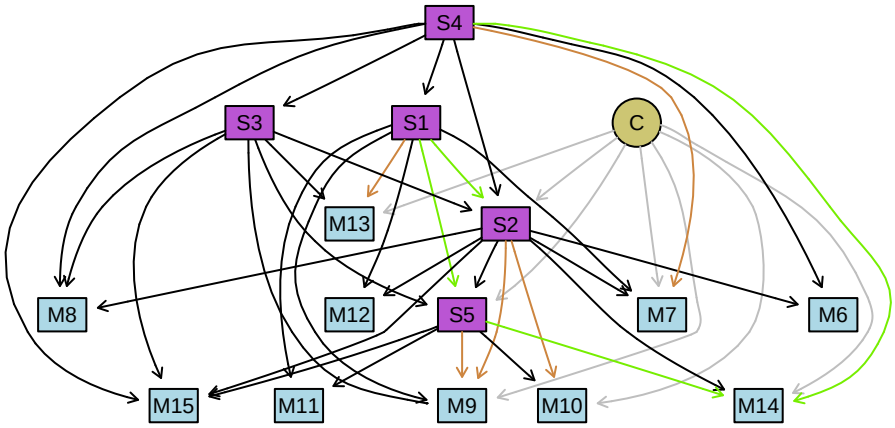
5.1 Tulokset oikealla malliperheellä

Jos opetusnäytteet ovat peräisin tunnetusta Bayes-verkolla esitetystä jakaumasta, pitäisi opetusalgoritmin ideaalitapauksessa pystyä löytämään täsmälleen oikea verkkorakenne. Yleensä näytteitä ei kuitenkaan ole tarpeeksi ja siksi löytyy pienestä näytemäärästä johtuvia näennäisiä riippuvuuksia. Toinen ongelma on opetus, joka käytännön syistä on toteutettu ahneena hakuna, minkä tiedetään olevan epäoptimaalista.

5.1.1 Tilanneriippuva graafinen malli

Ensimmäisessä kokeessa näytteet ovat peräisin kahdesta luokasta, joita vastaaviksi jakaumiksi on arvottu Bayes-verkot. Verkkojen rakenne kuuluu edellisessä luvussa kuvattuun malliperheeseen, eli samaan moduuliin kuuluvat geenit noudattavat samaa jakaumaa ja vain ne solmut, jotka kuuluvat säätelijämoduuleihin, voivat olla muiden moduulien isäsolmuja. Kummassakin verkossa on 400 solmua, jotka jakautuvat viiteen säätelijämoduuliin ja kymmeneen kohdemoduuliin siten, että jokaiseen säätelijämoduuliin kuuluu keskimääriin kuusi ja jokaiseen kohdemoduuliin 37 solmua. Solmut kuuluvat samoihin moduuleihin kummassakin luokassa, mutta kaarien välillä on hieman eroja. Verkot on muodostettu arpomalla ensin satunnaisesti (kaikki kaaret yhtä todennäköisiä) yksi verkko ensimmäiselle luokalle. Verkkoon tuli 39 kaarta. Toisen luokan verkko on muodostettu lisäämällä tai poistamalla ensimmäisestä verkosta sattumanvaraisesti 12 kaarta. Lopputuloksena verkoissa on eroja seitsemän moduulin

isäsolmuissa. Kuvaan 11 on piirretty molempien luokkien generoivat verkot yhtenä tilanneriippuvana verkkona. Kuvassa moduulien välisten kaarten värit kertovat kumpaan luokkaan ne kuuluvat. Jokaisen moduulin paikallisiksi jakaumiksi valittiin multinomijakauma, jonka parametrit arvottiin tasajakaumasta. Näiden seitsemän moduulin jakaumille on arvottu uudet parametrit. Osa riippuvuuksista siis pysyy samana ja ideaalitapauksessa menetelmä kytkee luokkamuuttujan vain niihin moduuleihin, joiden isäsolmut ovat erilaisia luokkien välillä.



Kuva 11: Luokkien generoivat verkot yhtenä tilanneriippuvana verkkona. Kummassakin luokassa esiintyvät kaaret ovat mustia, vain ensimmäisessä luokassa esiintyvät ruskeita ja vain toisessa esiintyvät vihreitä. Luokkasolmusta moduuleihin kulkevat kaaret ovat harmaita. Kuvassa on esitetty vain säätelijämoduulit (S) ja kohdemoduulit (M), ei yksittäisiä solmuja. Moduulien välillä on kaari, jos jokin isämoduulin solmuista vaikuttaa lapsimoduulin jakaumaan.

Verkoista generoitiin eri kokoisia opetusjoukkoja, joissa oli 30, 60, 100 tai 300 näytettä, jolloin näytteitä on 2–20 kertaa moduulien lukumäärä. Yhtä näytettä muodostettaessa valittiin aluksi luokka satunnaisesti. Sen jälkeen valitun luokan verkon koodaamasta jakaumasta arvottiin satunnaisvektori luvussa 3.3.3 esitellyllä tavalla.

Näytteitä käytettiin opettamaan useita erikokoisia verkkoja, joista pienimmässä oli kahdeksan ja suurimmassa kuusikymmentä moduulia. Verkkorakenteen ja sijoittelun priorina käytettiin tasajakaumaa ja paikallisten jakaumien säännöllisyyksille kuvauspituuteen perustuvaa prioria. Opetus toistettiin kymmenen kertaa kullakin verkon ja opetusjoukon koolla. Opetusta varten potentiaalisiksi säätelijöiksi valitiin kaikki solmut, jotka jommassakummassa generoivassa verkossa kuuluvat samaan moduuliin sellaisten solmujen kanssa, jotka ovat todellisuudessa isäsolmuja vähintään yhdelle moduulille. Menetelmän on siis periaatteessa mahdollista löytää täsmälleen oikea verkko.

Opittu verkko voidaan purkaa kahdeksi luokkakohtaiseksi verkoksi kiinnittämällä luokkamuuttuja vuorollaan kumpaankin mahdolliseen arvoonsa ja poistamalla verkosta kaaret, joita ei kyseisessä luokassa tarvita. Tulos on sitä parempi mitä lähempänä todellisia generoivia verkkoja ne ovat. Luokkakohtais-

ten verkojen kaaria verrattiin vastaavan luokan todellisen generoivan verkon kaariin käymällä läpi kaikki säätelijäsolmujen ja kohdemoduulin solmujen parit. Jos kahden solmun välillä on kaari sekä opitussa että generoivassa verkossa, niin kaarta nimitetään todelliseksi positiiviseksi kaareksi. Kaarta, joka on olemassa opitussa mutta ei generoivassa verkossa, kutsutaan vääräksi positiiviseksi. Vastaavasti solmuparit, joiden välillä ei ole kaarta opitussa verkossa, ovat todellisia negatiivisia, jossa myöskään generoivassa verkossa ei ole kaarta, tai vääriä negatiivisia, jos generoivassa verkossa on kaari solmujen välillä. Positiivisia ja negatiivisia kaaria laskettaessa yhdestä isäsolmusta lähtevää kaarta tarkastellaan siis erikseen jokaisen lapsisolmun suhteen, vaikka lapsisolmut kuuluisivatkin samaan moduuliin. Jos isäsolmusta moduuliin kulkeva kaari laskettaisiin vain kerran, niin isoonkin lapsimoduuliin kytketty kaari olisi yhtä merkitsevä kuin pienempään moduulin kytkeytyvä kaari.

Herkkyys ja tarkkuus

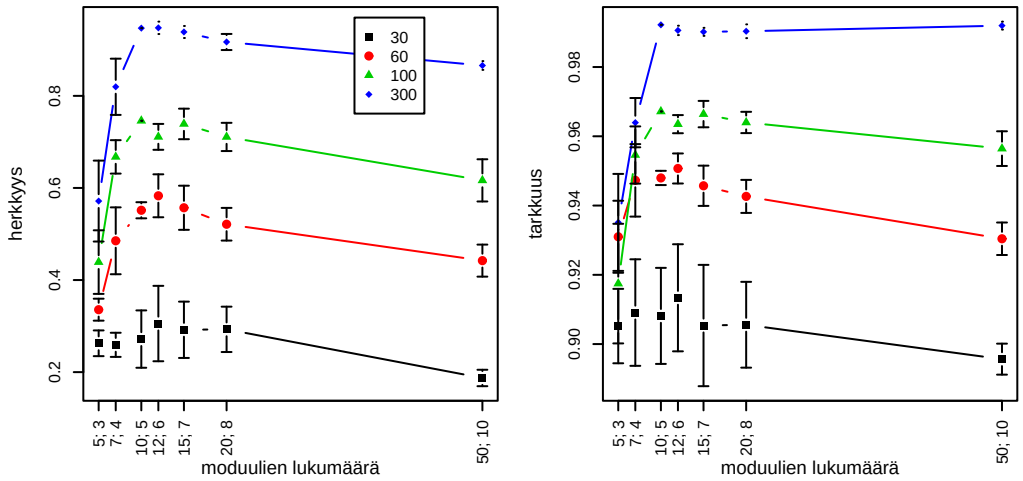
Opitun verkon samankaltaisuus todellisen verkon kanssa voidaan tiivistää kahdeksi kaarien vastaavuutta kuvaavaksi tunnusluvuksi. Herkkyys (sensitivity) on se osuus todellisen verkon kaarista, jotka menetelmä ennustaa oikein:

$$\text{herkkyys} = \frac{\#\text{oikeat positiiviset}}{\#\text{oikeat positiiviset} + \#\text{väärät negatiiviset}}.$$

Herkkyys ei yksinään vielä riitä kuvaamaan menetelmän hyvyyttä sillä, jos menetelmä palauttaisi aina täysin kytketyn verkon, herkkyys olisi yksi. Selvästikin tarvitaan myös puuttuvien kaarien määrän lukuunottava mittari. Tarkkuus (specificity) kertoo kuinka suurella todennäköisyydellä menetelmä jättää laittamatta kaaren sellaisen solmuparin välille, joiden välillä ei generoivassa verkossakaan ole kaarta:

$$\text{tarkkuus} = \frac{\#\text{oikeat negatiiviset}}{\#\text{oikeat negatiiviset} + \#\text{väärät positiiviset}}$$

Kuvassa 12 on esitetty menetelmän tuottamien verkkojen herkkyys ja tarkkuus. Luokkien positiiviset ja negatiiviset kaaret on summattu ja niistä on laskettu luokkien yhteiset herkkyydet ja tarkkuudet. Vaaka-akselilla on opetettujen verkkojen moduulien lukumäärät, ensimmäinen luku on kohdemoduulien ja jälkimmäinen luku säätelijämoduulien määrä. Kaikki käyrät näyttävät saavuttavansa huippunsa lähellä oikeaa moduulimäärää (5 säätelijämoduulia ja 10 kohdemoduulia). Jo sadalla näytteellä löytyy parhaimmillaan yli 70 % generoivan verkon kaarista tarkkuuden ollessa yli 96 %. Tosin sata näytettä tarkoittaa yli kuutta näytettä yhtä moduulia kohden, mikä lienee enemmän kuin ilmentymismittauksia yleensä on käytettävissä todellisuudessa. Jos moduuleita on liian paljon, tulokset ovat jonkin verran huonompia, mutta erityisesti suurimmalla näytemäärällä tulos pysyy erinomaisena vaikka moduuleita olisikin liikaa. Tämä tarkoittanee, että opetusalgoritmi pilkkoo moduuleita pienemmiksi mutta säilyttää niiden parametrit lähes samoina. Jos moduuleita on



(a) Herkkyys (b) Tarkkuus

Kuva 12: Herkkyys ja tarkkuus kertovat keinotekoisilla opetusaineistolla opetettujen verkkojen oikeiden ja väärien kaarien osuudet suhteessa todellisten generoivien verkkojen kaariin. Vaaka-akselilla on opetetun verkon moduulien lukumäärät; ensimmäinen luku kertoo kohdemoduulien ja jälkimmäinen säätelijämoduulien lukumäärät. Generoivassa verkossa oli 10 kohdemoduulia ja 5 säätelijämoduulia. Käyrät kuvaavat opetuksia, joissa opetusjoukon koko oli 30, 60, 100 tai 300 näytettä.

liian vähän, herkkyys on selvästi alhaisempi, mikä onkin ymmärrettävää koska silloin malli ei ole tarpeeksi joustava kyetäkseen esittämään generoivan verkon tarkasti. Tarkkuus on hyvin korkea joka verkolla, koska generoivat verkot ovat harvoja ja siksi myös opittavissa verkoissa on paljon oikeita negatiivisia kaaria.

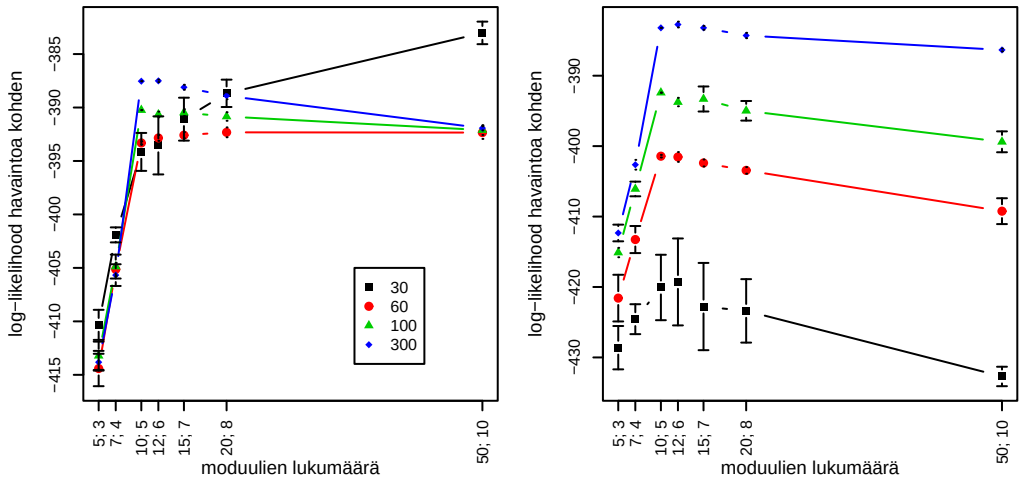
Marginaalinen uskottavuus

Yksittäisiä kaaria vertailevat herkkyys ja tarkkuus tarkastelevat opittua verkkoa paikallisesti. Uskottavuus sen sijaan riippuu koko verkon koodaamasta jakaumasta. Verkon marginaalisella uskottavuudella laskettuna näytejoukossa D tarkoitetaan paikallisten jakaumien parametrien yli integroitua uskottavuutta:

$$\log p(D|\mathcal{A},\mathcal{S}) = \log \int p(D|\mathcal{A},\mathcal{S},\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{A},\mathcal{S})\mathrm{d}\boldsymbol{\theta}.$$

Huomaa, että tämä on BD-pisteytysfunktio kaavasta (4) ilman rakenne- ja sijoitteluprioreita. Kuvassa 13(a) on opetusnäytteiden ja kuvassa 13(b) 5000 testinäytteen logaritminen marginaalinen uskottavuus normalisoituna näytteiden lukumäärillä kaikissa opetetuissa verkoissa. Testinäytteet on arvottu todellisista generoivasta verkosta, mutta niitä ei ole käytetty millään tavalla hyväksi opetuksessa.

Marginaalinen uskottavuus kasvaa lähestyttäessä oikeaa moduulimäärää (5 säätelijämoduulia ja 10 kohdemoduulia) ja pysyy lähes vakiona tai laskee hiljalleen kun moduuleita lisätään edelleen. Ainoa poikkeus on pienin opetusjoukko,



(a) Opetusnäytteet (b) Testinäytteet

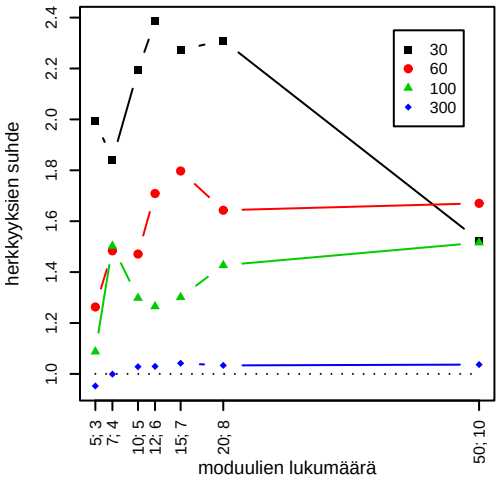
Kuva 13: Marginaaliset uskottavuudet keinotekoisella aineistolla. Merkinnät on selitetty kuvan 12 kuvatekstissä.

jolla uskottavuus kasvaa jatkuvasti mallin monimutkaistuessa. Kuvasta 13(b) näkyy, että pienimmällä näytemäärällä uskottavuuden varianssi on suurta ja opetus ylisovittuu, kun moduuleita on paljon. Isommilla näytemäärillä suurin uskottavuus saavutetaan oikealla moduulimäärällä. Jos opetettavassa verkossa on moduuleita hieman enemmän kuin generoivassa verkossa, niin tulos ei ole kovin paljon optimia huonompi, mutta 50 kohdemoduulin ja 10 säätelijämoduulin verkossa kaikkien muiden paitsi suurimmalla näytemäärällä opetetun verkon uskottavuus on pienentynyt jo selvästi. Nämä päätelmät tukevat herkkyydestä ja tarkkuudesta edellä tehtyjä havaintoja. Opetus- ja testinäytteiden absoluuttiset arvot näytettä kohden eivät ole suoraan vertailtavissa, koska niiden laskemisessa on käytetty normalisoimattomia jakaumia, mistä aiheutuu näytemäärästä riippuva skaalaus.

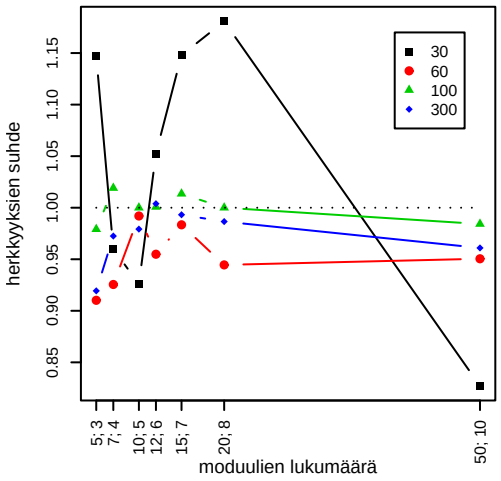
5.1.2 Moniverkkomalli

Tilanneriippuvan mallin vertailukohdaksi opetettiin oma verkko kummallekin luokalle samoilla opetusnäytteillä. Tätä kutsutaan moniverkkomalliksi. Siinä yhden verkon opetukseen on käytettävissä vain kyseisen luokan näytteet, eli noin puolet siitä näytemäärästä, joka oli käytettävissä tilanneriippuvan mallin opetukseen. Opittujen verkkojen kaaria verrattiin jälleen vastaavien generoivien verkkojen kaariin ja jokaiselle opetukselle laskettiin herkkyys ja tarkkuus samoin kuin edellä.

Kuvaan 14 on piirretty tilanneriippuvan mallin herkkyyksien keskiarvo kymmenessä opetuksessa suhteessa moniverkkomallin herkkyyksien keskiarvoon. Arvo yksi tarkoittaa, että mallit ovat herkkyydeltään yhtä hyviä, sitä suuremmat arvot merkitsevät, että tilanneriippuva malli on parempi. Kuvasta näkee, että tilanneriippuva malli saavuttaa suhteessa useita kymmeniä prosentteja



Kuva 14: Tilanneriippuvan mallin ja moniverkkomallin herkkyyksien suhde. Ykköstä suuremmat arvot tarkoittavat, että tilanneriippuva malli on parempi. Merkinnät on selitetty kuvan 12 kuvatekstissä.



Kuva 15: Tilanneriippuvan mallin ja moduuliverkon, jossa ei sallita luokkakohtaisia säätelijöitä, herkkyyksien suhde. Ykköstä suuremmat arvot tarkoittavat, että tilanneriippuva malli on parempi. Merkinnät on selitetty kuvan 12 kuvatekstissä.

paremman herkkyyden, jos opetusnäytteitä on vähän. Jos näytteitä on hyvin paljon, molemmat mallit ovat lähes yhtä hyviä. Tämä on linjassa sen hypoteesin kanssa, että pienillä näytemäärillä moniverkkomalli ylisovittuu pahemmin kuin tilanneriippuva malli, joka hyödyntää kaikkia näytteitä verkkojen yhteisten osien oppimisessa. Ainoat tapaukset, joissa moniverkkomalli on aavistuksen parempi, ovat sellaisia, joissa näytteitä on paljon ja opetettavasssa mallissa on liian vähän moduuleita, jolloin kumpikaan malli ei toimi kunnolla (vrt. kuva 12).

Mallien tarkkuuksista voi laskea samanlaisen suhteen, mutta koska tarkkuudet ovat myös moniverkkomallissa kaikissa opetuksissa melko korkeita, niin suhteelliset erot ovat vain muutamien prosenttien luokkaa, eikä tuloksia ole esitetty tässä.

5.1.3 Malli ilman paikallisten jakaumien säännöllisyyksiä

Toisena vertailukohtana käytetään moduuliverkkoa, jonka paikallisissa jakaumissa ei käsitellä luokkariippuvuuksia erityistapauksina. Muutoin käytettävän verkon rakenne vastaa tilanneriippuvaa mallia, eli verkon solmut on jaettu säätelijä- ja kohdemoduuleihin ja luokkamuuttuja on mukana ylimääräisenä solmuna. Tämä malli esittää siis luokkamuuttujan ja geenien yhteisjakaumaa. Vertailumallin moduulien jakaumat voivat riippua luokkamuuttujasta, mutta silloin jakauma riippuu aina samoista isäsolmuista joka luokassa toisin kuin tilanneriippuvassa mallissa, jossa eri luokissa voi olla osittain eri isäsolmut. Jos tällaista mallia haluttaisiin käyttää etsimään säätelyeroja luokkien välillä, pi-

täisi paikallisten jakaumien parametreille etsiä piste-estimaatit ja sen jälkeen testata jollain tilastollisella testillä onko jonkin luokan jakauma riippumaton jostain isäsolmusta. Tilanneriippuvassa mallissa säätelijöiden luokkakohtaisuus näkyy suoraan mallin rakenteesta, mikä on kiistaton etu.

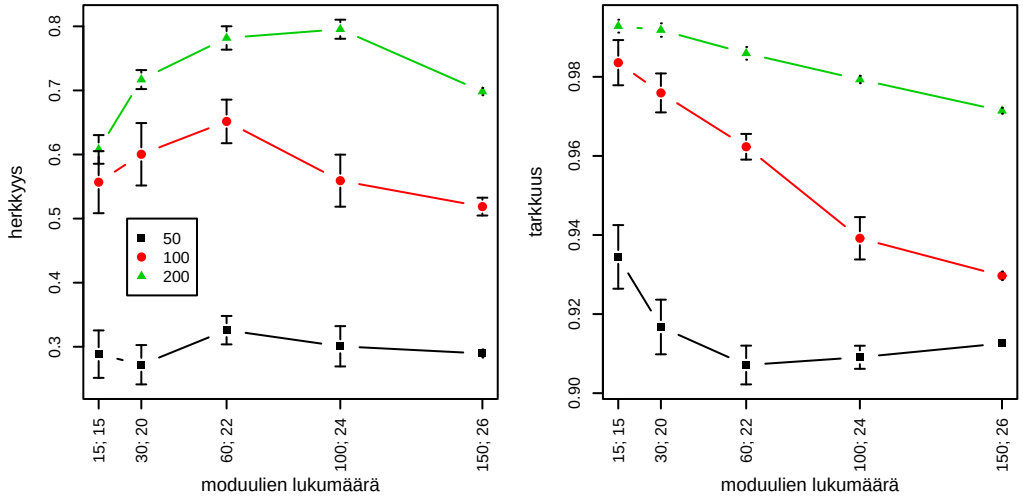
Malleja on verrattu jälleen laskemalla niiden suhteellinen herkkkyys. Erot ovat vain muutamia prosentteja muilla paitsi kaikkein pienimmällä opetusjoukolla, jossa vaihtelu on suurta erikokoisten moduulien välillä (kuva 15). Koska tilanneriippuva malli saavuttaa suurinpiirtein yhtä hyvän opetustuloksen kuin verkko ilman luokkamuuttujan erityiskäsittelyä, mutta toisaalta luokkakohtaisten säätelyvuorovaikutusten tulkitseminen opitussa verkossa on huomattavasti helpompaa tilanneriippuvalla mallilla (ks. luku 4.4), niin sen käyttö on perusteltua.

5.2 Tulokset todenmukaisemmalla generointiprosessilla

Todellisuudessa mitattujen ilmentymisnäytteiden jakauma ei kuulu käytettyyn malliperheeseen. Menetelmää voidaan kuitenkin käyttää, jos oletetaan että todellinen jakauma on lähellä malliperhettä. Silloin opetuksen pitäisi valita malliperheestä verkkorakenne, joka mahdollisimman hyvin approksimoi todellista jakaumaa. Jos ero ei ole kovin suuri, niin opittu verkko on lähellä oikeaa ja sitä voidaan käyttää jatkopäätelmien perustana. Realistisemman verkkorakenteen vaikutuksen testaamiseksi toistettiin samanlainen opetus kuin edellä, mutta opetusnäytteet generoiva verkko arvottiin SynTReN-verkkogeneraattorista (ks. luku 3.3.3 ja viite [48]).

SynTReN-ohjelmalla arvottiin 200 geenin verkko. Arvottu verkko muunnettiin modulaariseksi Bayes-verkoksi yhdistämällä kaikki geenit, joilla on samat isäsolmut samaan moduuliin. Geenit, joilla ei ole yhtäkään isäsolmua, laitettiin kuitenkin jokainen yksinään omaan moduuliinsa, koska ei ole realistista olettaa, että kaikki geenit, joilla ei ole säätelijöitä noudattaisivat samaa jakaumaa. Tällä tavalla syntyi 59 ”luonnollista” moduulia, joihin kuului keskimäärin 3,4 geeniä. SynTReNin tuottama verkko on siis huomattavasti hajanaisempi mitä Segal et al. [43] olettavat geenien säätelyverkkojen olevan, koska he arvioivat yhden moduulin kooksi keskimäärin noin 50 geeniä. Yhtenä syynä tähän on se, että johtuen SynTReNin tavasta generoida verkkoja kaikki geenit ovat aina kytketty toisiinsa ainakin yhdellä polulla, mikä ei välttämättä ole biologisesti perusteltua. Lisäksi Segal et al. tarkoittanevat moduulilla yhteen solun toimintoon liittyviä geenejä, vaikka niitä kaikkia ei säädelläkään täsmälleen samalla tavalla.

Toiselle luokalle muodostettiin verkko jälleen tekemällä joitain satunnaisia muutoksia ensimmäisen luokan verkkoon. Molempien verkkojen moduulien multinomijakaumille arvottiin parametrit tasajakaumasta ja verkoista gene-

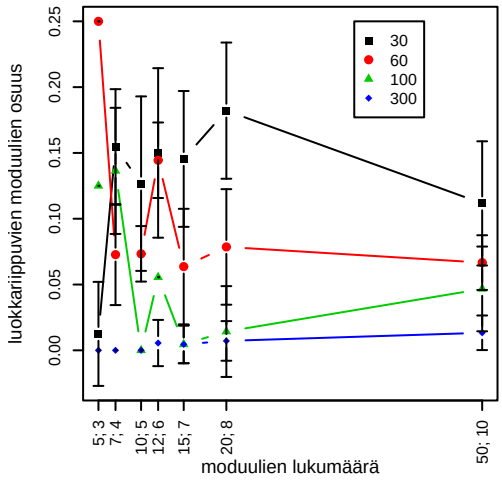


(a) Herkkyys (b) Tarkkuus

Kuva 16: Herkkyys ja tarkkuus SynTReN-ohjelmalla muodostetusta verkosta generoiduilla näytteillä. Merkinnät on selitetty kuvan 12 kuvatekstissä.

roitiin näytevektoreita samalla tavalla kuin edellä. Opetuksessa potentiaalisten säätelijöiden joukko koostui sekä todellisista säätelijöistä, jotka ovat toisen solmujen isäsolmuja jommassakummassa verkossa, että solmuista, jotka eivät oikeasti säätele muita solmuja. Näitä ”ylimääräisiä” säätelijöitä oli noin viidennes potentiaalisista säätelijöistä. Ne tekevät opetuksesta realistisemman, koska todellisuudessakaan kaikki potentiaaliset säätelijät eivät välttämättä toimi kaikissa tilanteissa. Geeni lasketaan potentiaaliseksi säätelijäksi, jos on olemassa edes jotain merkkejä siitä, että se jossain tilanteessa voisi toimia säätelijänä, esimerkiksi jos sen DNA-sekvenssissä on samankaltaisuuksia tunnettujen säätelijöiden kanssa.

Opetettujen verkkojen kaaria verrattiin generoiviin verkkoihin laskemalla niille herkkyys ja tarkkuus. Tulokset on esitetty kuvassa 16. Tarkkuus on edelleen korkea, mutta herkkyydessä silmiinpistävintä on se, että jos näytteitä on paljon, niin huippukohta saavutetaan moduulimäärällä, joka on ”luonnolista” määrää suurempi. Esimerkisi kahdensadan näytteen käyrä on korkeimmillaan, kun verkossa on 100 kohdemoduulia ja 24 säätelijämoduulia, eli keskimäärin yhdessä moduulissa on vain noin 1,6 geeniä. Tarkkuus kuitenkin laskee hiljalleen moduulimäärän kasvaessa, eli opetus näyttää lisäävän verkkoon paljon kaaria, joista osa on vääriä. Osa syynä tähän on varmasti SynTReN-verkon hajanaisuus ja ylimääräiset säätelijät, jotka menetelmän täytyy sovittaa säätelijämoduuleihin. Olisikin mielenkiintoista kokeilla opetusta useilla säätelijämoduulimäärillä pitäen kohdemoduulien määrä vakiona, mutta parametrien määrä ja opetukseen kuluva aika kasvaisivat silloin huomattavan suuriksi. Suurimmalla kokeilluilla moduulimäärällä kaikkien opetusten herkkyydet ovat jo pienentyneet maksimistaan, eli tässäkin tapauksessa moduuliverkko on kuitenkin parempi kuin normaali Bayes-verkko, jossa jokaiselle geenille olisi oma moduuli.



Kuva 17: Opituissa verkoissa luokkamuuttujasta (virheellisesti) riippuvien moduulien osuus kaikista moduuleista, kun kummankin luokan näytteet ovat todellisuudessa peräisin samasta jakaumasta. Merkinnät on selitetty kuvan 12 kuvatekstissä.

5.3 Oikeiden luokkariippuvuuksien osuuden empiirinen estimointi

Tilanneriippuvan mallin opetusalgoritmin tavoitteena on asettaa luokkamuuttuja vain sellaisten moduulien isäsolmuksi, jotka oikeasti käyttäytyvät eri tavalla eri luokissa, eli joiden jakauma riippuu luokasta. Äärellisen näytemäärän takia opetuksessa voi kuitenkin löytyä näennäisiä luokkariippuvuuksia siellä-kin, missä niitä ei todellisuudessa ole, tai osa riippuvuuksista voi jäädä löytymättä. Jotta saataisiin käsitys siitä, kuinka paljon olemattomia luokkariippuvuuksia menetelmä löytää, toistettiin samankaltainen opetus kuin edellä, mutta nyt kummankin luokan opetusnäytteet arvottiin samasta jakaumasta. Koska jakaumat ovat samat, ei yhdenkään moduulin paikallinen jakauma todellisuudessa riipu luokasta, eli luokkamuuttujalla ei pitäisi olla yhtäkään lasta. Generoivana jakaumana käytettiin samaa jakaumaa kuin aliluvun 5.1 ensimmäiselle luokalle eli se kuuluu malliperheeseen.

Kuvaan 17 on piirretty luokasta riippuvien moduulien osuus kaikista moduuleista jokaisessa opetuksessa. Alle 25 prosenttia moduleista riippuu virheellisesti luokasta. Kuvasta näkyy, miten opetusnäytteiden lisääminen pienentää virheellisten luokkariippuvuukisen määrää. Suurimmalla opetusjoukon koolla vain pari prosenttia moduuleista riipppuu luokasta. Tulosta voi pitää jonkinlaisena arviona siitä, kuinka suuri osa menetelmän ennustamista luokkariippuvuuksista itseasiassa on vääriä eri kokoisilla opetusjoukoilla.

Koska näyttää siltä, että väärien luokkariippuvuuksien määrä lähestyy nollaa opetusnäytteiden määrän kasvaessa, voi menetelmää periaatteessa käyttää testaamaan tulevatko eri luokkien näytteet itseasiassa samasta jakaumasta, jos näytteitä on riittävästi. Jos opetetussa verkossa on vain hyvin vähän kytkentöjä luokkasolmuun, niin luokkien jakaumat ovat todennäköisesti samat.

5.4 Yhteenveto

Riittävän suurella näytemäärällä tilanneriippuva graafinen malli löytää todellisen generoivan verkon melko hyvällä tarkkuudella kunhan opittavassa verkossa on vähintään yhtä paljon moduuleita kuin oikeassa verkossa. Moduulien määrän kasvattaminen hieman todellista suuremmaksi huonontaa tuloksia vain vähän.

Tulokset ovat erityisesti pienillä näytemäärillä paljon parempia kuin opetettaessa kummallekin luokalle oma verkko. Tämä osoittaa, että molempien luokkien käsitteleminen samassa verkossa vähentää selvästi ylisovittumista, koska yhteisten verkon osien oppimiseen voidaan käyttää kaikkia näytteitä. Ilman paikallisten jakaumien säännöllisyyksiä opittu verkko on suurinpiirtein yhtä hyvä kuin luokkamuuttujaa erityistapauksena käsittelevä verkko. Tilanneriippuvan verkon etuna on luokkariippuvien säätelyvuorovaikutusten huomattavasti helpompi tulkittavuus.

Luku 6

Sovellus stressaavien olosuhteiden ilmentymismittauksiin

Edellisessä luvussa tilanneriippuva graafinen malli opetettiin tunnetusta jakaumasta keinotekoisesti tuotetuilla näytteillä. Silloin oppimisalgoritmin löytämän verkon rakennetta voitiin helposti verrata tunnettuun todelliseen verkkoon. Tämä on tietenkin hyvin keinotekoinen asetelma. Käytännössä verkko opetetaan aitojen ilmentymismittausten avulla, jolloin oikeaa verkkoa ei tietenkään tunneta. Opituille säätelyvuorovaikutuksille voitaisiin hakea vahvistusta kirjallisuudesta, mutta vain murto-osalle säätelysuhteita on raportoitu kokeellisia tuloksia. Vielä pahempi ongelma on se, että kirjallisuuteen vertaamalla ei ole mahdollista selvittää onko menetelmän löytämä vuorovaikutus uusi, ennalta tuntematon biologinen löydös vai väärä positiivinen. Näiden syiden vuoksi tuloksen paikkansapitävyyden arviointiin täytyy käyttää epäsuoria keinoja. Tässä luvussa verkkoa käytetään etsimään säätelyeroja stressiä aiheuttavien ja normaaliolosuhteiden välillä ja tulosta verrataan biologista tutkimusta kokoaviin tietokantoihin.

6.1 Suoritetut kokeet

Käytetty mittausaineisto

Gasch et al. [18] ovat mitanneet leivontahiivan geenien ilmentymistasoja olosuhteissa, joiden tiedetään käynnistävän hiivan soluissa sopeutumisohjelman, jonka tarkoituksena on auttaa soluja sopeutumaan ympäristöolosuhteisiin. Tällaisia käsittelyitä ovat esimerkiksi lämpötilan nopea muutos, altistaminen myrkyllisille aineille tai epäoptimaaliset kasvuolosuhteet, kuten liian alhainen typpitaso ruokaliuoksessa. Mittaus on toistettu jokaisessa olosuhteessa useilla eri ajanhetkillä.

Hiivassa on noin 800 geeniä, jotka reagoivat stressiä aiheuttaviin ympäristö-

olosuhteisiin. Gasch et al. kutsuvat niitä ESR-geeneiksi (environmental stress response). Ne jakautuvat geeneihin, joiden aktiivisuus kasvaa (aktivoitu-ESR), ja geeneihin, joiden aktiivisuus laskee (inhiboitu-ESR) stressissä. ESR-geenit ohjaavat hiivan yleistä puolustautumisohjelmaa vihamielisiä olosuhteita vastaan. Osa ESR-geeneistä ohjaa vain jonkin tietyn ympäristömuutoksen, kuten lämpöshokin, sopeutumisohjelmaa.

Stressin ja normaalitilan säätelyerojen löytämiseksi opetettiin tilanneriippuva verkko käyttäen toisena luokkana Gaschin ilmentymismittauksia ja toisena luokkana hiivan normaalitilan ilmentymisprofiileja. Koska tarkoituksena on etsiä yleistä stressivastetta, kaikki eri stressaavien olosuhteiden näytteet on niputettu yhteen, jolloin stressinäytteitä kertyy 80 kappaletta. Normaalitilanteen mittaukset ovat alunperin solusyklin tutkimiseen tarkoitettuja mittauksia [47]. Solusykli on solun kasvuvaiheessa läpikäymä prosessi. Sykli alkaa solun jakauduttua kahdeksi ja valmistelee solun uuteen jakautumiseen. Solusyklin tutkimista varten mittauksissa kaikki kannan solut on pyritty synkronisoimaan samaan vaiheeseen. Synkronisointi hajaantuu, kun mittauksia jatketaan usean solusyklin ajan. Koska synkronisointi ei ole soluille luonnollista, tätä työtä varten solusykliprofiileista jätettiin pois osa alkupään mittauksista, joissa synkronisointi on tiukimmillaan. Solusyklinäytteitä jäi poistamisen jälkeen opetusta varten 53 kappaletta.

Opetusnäytteitä ei ole riittävästi kaikista hiivan yli 6000 geenistä muodostuvan verkon opetukseen. Sen vuoksi opetettavaan verkkoon valittiin 1268 geeniä, joilta puuttuu alle 20 mittausarvoa ja joiden ilmentymisaktiivisuus poikkeaa paljon normaalitilanteen aktiivisuudesta vähintään yhdessä stressimittauksessa. Tällä kriteerillä mukaan tuli valittua suurin osa kaikista ESR-geeneistä (582 ESR-geeniä 868:sta), joten valintakriteerit ovat järkeviä stressireaktioiden mallintamisen kannalta. Valittujen geenien mittausten jäljelle jääneet puuttuvat arvot korvattiin geenikohtaisilla keskiarvoilla. Potentiaaliset säätelytekijät valittiin samalla tavalla kuin Segal et al. (ks. luku 2.5 ja viite [43]). Karsinnan jälkeen jäljelle jää 69 potentiaalisita säätelijää, joista 18 on ESR-geenejä.

Opetettavassa verkossa oli 15 säätelijämoduulia ja 25 kohdemoduulia. Säätelijämoduuleihin kuuluu siis keskimäärin noin neljä säätelijää ja kohdemoduuleihin keskimäärin 50 geeniä, mikä on Segalin [43] mukaan biologisesti uskottava yhden moduulin geenimäärä.

Diskretointi

Ilmentymisprofiilit ovat reaaliarvoisia logaritmisia ilmentymistasoja suhteessa referenssitilan ilmentymistasoon. Vaikka Bayes-verkoilla pystytään yleisessä tapauksessa käsittelemään myös jatkuva-arvoisia muutujia, kykenee tässä työssä esiteltävä malli käyttämään vain diskreettejä muuttujia. Siksi ilmentymisprofiilit on aluksi diskretoitava.

Diskretoinnissa todennäköisesti häviää informaatiota, mutta joissain suhteissa

siitä voi olla myös apua. Diskretointi voi vähentää kohinaa, jota ilmentymismittauksissa tiedetään olevan. On myös joitain todisteita, että monilla säätelijöillä on vain muutamia toimintapisteitä (kuten pois päältä/alhainen/voimakas) joiden välillä geenin ilmentyminen ei muutu tasaisesti [20, kolmas luku], jolloin diskretointi tuottaa itseasiassa tarkemman kuvan todellisesta aktiivisuudesta. Jatkuva-arvoisia muuttujia käytettäessä paikallisten jakaumien muodoksi voitaisiin valita esimerkiksi regressiopuut normaalijakautuneilla lehdillä [43] tai epäparametrinen regressio [27], mutta ei ole olemassa mitään syytä miksi juuri nämä olisivat hyviä malleja ilmentymiselle. Diskretoiduille arvoille multinomijakauma on luonnollinen valinta, koska se mahdollistaa mielivaltaiset stokastiset kombinatoriset riippuvuudet muuttujien välillä ja toisaalta tiedetään, että monet säätelijät vaikuttavat vain yhdessä muiden säätelijöiden kanssa. Monet jatkuvia muuttujia käyttävät menetelmät olettavat, että isäsolmujen vaikutus summautuu lineaarisesti, mikä ei välttämättä pidä paikkaansa ilmentymissäätelyssä.

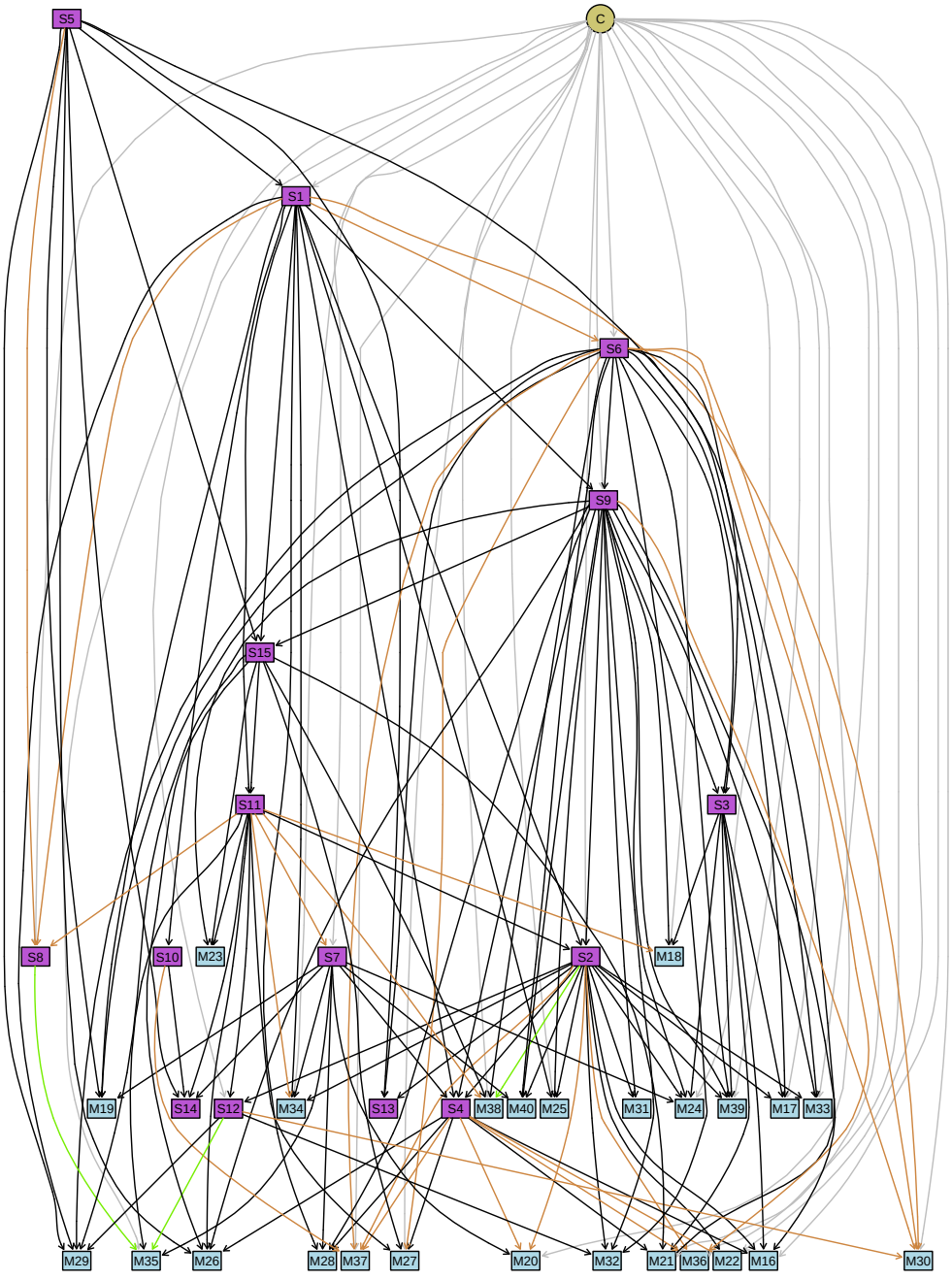
Jatkuva-arvoisia muuttujia diskretoitaessa täytyy valita diskretointitasojen lukumäärä. Jos tasoja on liian vähän, malli ei pysty esittämään monimutkaisia riippuvuuksia ja diskretointi kadottaa informaatiota. Liian hienojakoinen jako puolestaan kasvattaa estimoitavien parametrien määrää. Yu et al. [52] ovat testanneet empiirisesti mm. diskretointitasojen määrän vaikutusta Bayes-verkon oppimiseen. Heidän mukaansa kolmella tasolla saavutetaan parhaat tulokset. Sen vuoksi tässäkin työssä käytettiin kolmea diskretointitasoa. Niiden voi ajatella edustavan referenssitasoa alhaisempaa, referenssiä vastaavaa ja referenssiä korkeampaa ilmentymisaktiivisuutta.

Diskretointitasojen rajoiksi on valittu ilmentymisaktiivisuuden kaksinkertaistuminen tai puolittuminen referenssiin nähden. Rajojen kiinnittäminen etukäteen on yksinkertaisin mahdollinen diskretointikeino, mutta luultavasti epäoptimaalinen. Se ei esimerkiksi huomioi, että joidenkin geenien ilmentymisaktiivisuus voi normaalitilanteessa vaihdella paljon enemmän kuin toisten.

6.2 Tulokset

Verkko opetettiin kymmenkertaisella uudelleenkäynnistettävällä ahneella haulla, jossa yhden opetuksen jälkeen verkkoon tehtiin hieman muutoksia ja haku käynnistettiin uudelleen. Tarkoituksena on, että uudelleenkäynnistys auttaa hakua löytämään paremman lopputuloksen välttämällä paikalliset minimit. Ohjelma on toteutettu R-ohjelmointikielellä. Yksi opetus 2,2 GHz AMD Opteron työasemalla vei noin tunnin, joten yhteensä kaikki uudelleenkäynnistykset kestivät noin yksitoista tuntia.

Toisin kuin keinotekoisesti generoitujen näytteiden tapauksessa todellisilla ilmentymisnäytteillä opetetulle verkolle ei tunneta oikeaa generoivaa verkkoa. Siksi tässä tapauksessa ei voida samalla tavalla tarkistaa ovatko opitun verkon kaaret oikeita, mutta jonkinlainen biologinen validaatio on mahdollista.



Kuva 18: Opitut moduulit ja moduulien väliset kaaret. Ylhäällä on luokkamuuttuja, säätelijämoduulien ensimmäinen kirjain on S ja kohdemoduulien M. Luokkamuuttujasta moduuleihin osoittavat kaaret on värjätty harmaalla, vain stressiluokkaan kuuluvat kaaret ruskealla ja vain solusykliluokkaan kuuluvat kaaret vihreällä. Kaaret, jotka kuuluvat molempiin luokkiin, ovat mustia.

Opitun verkon 40 moduulista 24 on sellaisia, joiden paikallinen jakauma riippuu luokkamuuttujan arvosta. Niistä 16:ssa on oikeasti erilaiset isäsolmut eri luokissa ja lopuissa 8:ssa on samat isäsolmut, mutta eri jakaumat. Stressiluokassa moduuleilla on keskimäärin 3,5 ja normaalitilassa 2,95 säätelijää. Epätasapaino on biologisesti perusteltavissa, koska solussa on todennäköisesti käynnissä paljon perusprosesseja sekä stressaavissa että normaalitilanteessa ja stressitilantessa on niiden lisäksi käynnistynyt sopeutumisreaktioita. Opittu verkko on piirretty kuvaan 18. Yksityiskohtaisuuden välttämisen takia kuvassa on vain moduulit, ei yksittäisiä solmuja. Kahden moduulin välille on piirretty kaari, jos vähintään yksi ensimmäisen moduulin solmuista on jälkimmäisen moduulin isäsolmu. Vain toiseen luokista kuuluvat kaaret on väritetty eri väreillä. Liitteessä A on tarkemmat tiedot jokaisesta moduulista.

Opittujen moduulien biologisen mielekkyyden varmistamiseksi jokaisen moduulin geenejä verrattiin geeniontologialuokkiin (ks. luku 2.4). Jokaisen moduulin geenejä verrattiin kaikkiin GO-luokkiin laskemalla yhteensopivuutta mittaava p-arvo. Geenit voidaan jaotella kahdella tavalla; ne joko kuuluvat käsiteltävään moduuliin tai eivät kuulu ja toisaalta ne voidaan jakaa tiettyyn GO-luokkaan kuuluviin ja kuulumattomiin. Fisherin tarkka testi laskee hypergeometrisen p-arvon, joka mittaa sitä kuinka paljon nämä kaksi jaottelua riippuvat toisistaan. Pieni p-arvo tarkoittaa, että nollahypoteesille, eli riippumattomuudelle, ei löydy tukea havainnoista. Toisin sanoen pieni p-arvo siis tarkoittaa, että tarkasteltavaan moduuliin kuuluu enemmän tarkasteltavan GO-luokan geenejä kuin olisi odotettavissa, jos geenit olisi jaettu moduuleihin täysin satunnaisesti.

Koska jokaista moduulia verrataan jokaiseen GO-luokkaan, tulee testejä tehtyä valtava määrä, joten on odotettavissa, että pelkästään sattumalta osa testeistä osoittautuu merkitseviksi. Yksinkertaisin tapa korjata p-arvot siten, että niiden perusteella voidaan edelleen kontrolloida virheellisesti merkitseviksi sanottavien testien osuutta, on Bonferroni-korjaus. Siinä jokainen p-arvo kerrotaan testien määrällä ja merkitseviksi sanotaan, niitä testejä, joissa korjattu p-arvo on pienempi kuin jokin etukäteen päätetty raja-arvo, esimerkiksi 0,05. Bonferroni-korjaus on hyvin konservatiivinen, eli se jättää pois osan tuloksista, jotka oikeasti ovat merkitseviä. Toinen ongelma on GO-luokkien riippuvuudet. GO-termit muodostavat puun, jossa geenit, jotka liittyvät lapsitermiin, liittyvät aina myös isätermiin. Tämän takia termien rikastumistestit eivät ole riippumattomia kuten Bonferroni ja useimmat muut korjaukset olettavat. Riippuvuudet huomioivia p-arvojen korjausmenetelmiä on kehitelty (esim. [2]), mutta ne ovat melko monimutkaisia. Liitteessä A on tyydytty luettelemaan kaikki GO-luokat, joiden korjaamattomat p-arvot ovat alle 0,05. Niistä voi valita vain osan kärkipäästä. Mitä pienempää raja-arvoa käytetään, sitä varmemmin löydetään todellisia rikastumia.

Vain kahteen viidestätoista säätelijämoduulista liittyy vähintään yksi rikastunut GO-luokka. Kohdemoduuleista noin puolessa (11/25) on rikastuneita GO-luokkia. Molempien rikastuneiden säätelijämoduulien GO-termit liittyvät sääntelyyn. Moduulin S12 rikastuneet termit viittaavat proteiinikinaasien sää-

telyyn. Kinaasit ovat entsyymejä, jotka sitovat tietyntyyppisiä molekyylejä toisiinsa. Moduuli S15 GO-termi on *transkriptiotekijöiden aktiivisuus* (engl. *transcription factor activity*), joten moduuliin kuuluu geenejä, jotka vaikuttavat toisten geenien ilmentymisaktiivisuuteen transkriptiovaiheessa. Rikastumien vähyys johtuu Bonferroni-korjauksen konservatiivisuudesta. Lisäksi useissa säätelijämoduuleissa on vain muutama geeni, minkä takia merkitsevien tuloksien saaminen millä tahansa testillä on epätodennäköistä.

Moduulien ja GO-luokkien p-arvot mittaavat opitun verkon yleistä biologista mielekkyyttä, mutta ne eivät kerro siitä kuinka hyvin verkko erottelee stressin ja normaalitilan säätelyvuorovaikutukset toisistaan, mikä oli opetuksen tavoite. Epäsuoraa todistusaineistoa tästä antaa ESR-geenien jakautuminen moduuleihin. Yhdessäkään moduulissa ei ole sekä aktivoitu-ESR että inhiboitu-ESR-geenejä, kuten sopii odottaa. ESR-geenit ovat jakautuneet hyvin epätasaisesti moduuleihin. Kaikista 40 moduulista 12 ei sisällä lainkaan ESR-geenejä. Toisaalta kuuden moduulin geeneistä vähintään 90 prosenttia on ESR-geenejä. Kaikista geeneistä ESR-geenejä on noin 46 prosenttia. Tämä viittaa vahvasti siihen suuntaan, että vain osa moduuleista liittyy stressiin.

Opitussa verkossa on 143 kaarta (poislukien luokkasolmusta muihin solmuihin kulkevat kaaret), näistä 25 esiintyy vain stressiluokassa, 3 vain solusykliluokassa ja 115 molemmissa. Kaarien merkittävyyden arvioimiseksi niihin liittyviä säätelijöistä haettiin informaatiota *Saccharomyces Genome Database* -tietokannasta [12]. SGD kerää yhteen useista eri lähteistä tutkimustietoa hiivan geeneistä. Jokaisesta geenistä kerrotaan mm. geeniontologialuokka, kirjallisuudessa koottuja mainintoja geenin tehtävistä solussa ja muita tietoja, joita on saatavilla. Verkon tulkinnan kannalta mielenkiintoisimmat tiedot ovat säätelijägeenien kirjallisuudessa ennustetut roolit. Niistä 25 kaaresta, jotka opitussa verkossa kuuluvat pelkästään stressiluokkaan, 20:n isäsolmu on SGD:n mukaan tunnettu stressireaktioiden säätelijä. Kaikkien kolmen pelkästään solusykliluokkaan kuuluvan kaaren säätelijät on SGD:ssä yhdistetty solusyklin säätelyyn. Kumpaankin luokkaan liittyvien 115 kaaren säätelijöistä SGD:ssä on 76:n kohdalla mainittu kytkös joko stressiin tai solusykliin ja 16 liittyy molempiin. Koska ainakin osa näistä kaarista on solun perustoimintojen säätelyvuorovaikutuksia, jotka ovat käynnissä kokoajan, niin ei voi olettaakaan, että ne kaikki liittyisivät stressin tai solusyklin säätelyyn. Näiden validointi on vaikeampaa. Kaikenkaikkiaan tulokset ovat varsin hyviä. Melkein kaikki säätelijät kaarissa, joiden menetelmä ennustaa kuuluvan vain toiseen luokista, on kirjallisuudessa yhdistetty juuri kyseiseen luokkaan.

Luku 7

Johtopäätökset

Bayes-verkkojen käyttö geenien ilmentymisen säätelyn mallintamiseen perustuu yksinkertaiseen vastaavuuteen matemaattisen ja biologisen mallin välillä. Bayes-verkossa solmujen paikalliset jakaumat riippuvat yleensä vain parista muuttujasta ja toisaalta yhden geenin ilmentymistä säätelee yleensä vain muutama proteiini, joten jos Bayes-verkon solmut vastaavat geenejä, niin kaarien voi ajatella vastaavan säätelysuhteita. Todennäköisyyslaskentaan perustuvina Bayes-verkot pystyvät käsittelemään kohinaa, jota ilmentymismittauksissa tunnetusti on. Verkon oppimiseksi tehdyistä ilmentymismittauksista täytyy määritellä funktio, joka kertoo kuinka hyvin annettu verkko kuvaa havaintojen empiiristä jakaumaa. Opetus voidaan nähdä diskreettinä optimointitehtävänä, jossa on tarkoituksena etsiä mitan maksimoiva verkko. Yleensä opetus suoritetaan ahneella haulla, joka etenee tekemällä verkkoon pieniä muutoksia ja hyväksymällä niistä parhaan.

Aiemmin julkaistut Bayes-verkkomenetelmät olettavat että kaikki näytteet ovat peräisin samasta jakaumasta. Jos niitä haluttaisiin käyttää säätelyn olosuhde-erojen mallintamiseen, täytyisi jokaisen olosuhteen mittauksille muodostaa erillinen verkko ja verrata niitä keskenään. Tällöin kuitenkin yhtä verkkoa kohden olisi käytettävissä vähemmän opetusnäytteitä ja myös ne säätelysuhteet, jotka todellisuudessa pysyvät samanlaisia eri olosuhteissa, estimoitaisiin erikseen kummallekin tapaukselle, mikä kasvattaa ylisovittumisen riskiä ja tekee tuloksista epäluotettavampia.

Tässä työssä on laajennettu aiemmin esiteltyjä Bayes-verkkomenetelmiä käsittelemään olosuhteesta riippuvia isäsolmuja. Uusi menetelmä löytää automaattisesti ne osat säätelyverkosta, joissa on eroja luokkien välillä, ja käyttää niiden oppimiseen vain oikean luokan havaintoja. Muuttumattomana pysyvien verkon osat opitaan kaikkien näytteiden avulla, jolloin tulokset ovat luotettavampia, kuin käytettäessä erillisiä verkkoja joka luokalle.

Keinotekoisesti muodostetuilla opetusaineistoilla suoritettujen kokeiden tulokset osoittavat, että jos näytteitä on tarpeeksi suhteessa moduulien määrään, niin uusi menetelmä löytää verkon, joka vastaa varsin tarkasti oikeaa. Lisäksi

osoitettiin, että tilanneriippuva malli tuottaa lähempänä todellista olevia verkkoja kuin erillisen verkon opettaminen joka luokalle. Tämä oli odotettavissa, koska tilanneriippuva malli pystyy käyttämään kaikkien luokkien näytteitä hyväksi opittaessa sellaisia verkon osia, jotka pysyvät samoina eri luokissa.

Jos tarkoituksena on etsiä pelkästään säätelyn eroja, niin selvästikin resurssit tulisivat tehokkaammin käytettyä, jos voitaisiin keskittyä vain eroihin ja jätettäisiin muuttumattomana pysyvät säätelysuhteet kokonaan mallintamatta. Ikävä kyllä ei ole lainkaan selvää miten tämä tulisi käytännössä toteuttaa, jos ei etukäteen tiedetä mitkä osat säätelyverkosta pysyvät samanlaisina eri tilanteissa. Tässä työssä esitellyn verkon täytyy mallintaa kaikkia säätelysuhteita, jotta opetusalgoritmi pystyy etsimään ne kohdat, joissa on eroja mittausten välillä.

Menetelmää voisi jatkokehittää arviomaan löydetyn verkon kaarien ja moduulien luotettavuutta. Toinen mielenkiintoinen lisäys voisi olla säätelijöiden etsiminen verkon oppimisen yhteydessä sen sijaan, että potentiaaliset säätelytekijät pitää kiinnittää etukäteen. Myös informatiivisten rakenne- ja sijoitteluprioorien käyttö voisi laajentaa menetelmän käyttökelpoisuutta.

Viitteet

[1] Tatsuya Akutsu, Satoru Kuhara, Osamu Maruyama ja Satoru Miyano. Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. Kirjassa *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, ss. 695–702, Philadelphia, PA, 1998. Society for Industrial and Applied Mathematics.

[2] Adrian Alexa, Jörg Rahnenführer ja Thomas Lengauer. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. *Bioinformatics*, 22(13):1600–1607, 2006.

[3] James E. Bailey. Lessons from metabolic engineering for functional genomics and drug discovery. *Nature Biotechnology*, 17:616–618, 1999.

[4] G. Octo Barnett, Kathleen T. Famiglietti, Richard J. Kim, Edward P. Hoffer ja Mitchell J. Feldman. DXplain on the Internet. Kirjassa *Proceedings of the American Medical Informatics Association*, ss. 607–611, 1998.

[5] Katia Basso, Adam A. Margolin, Gustavo Stolovitzky, Ulf Klein, Riccardo Dalla-Favera ja Andrea Califano. Reverse engineering of regulatory networks in human B cells. *Nature Genetics*, 37(4):382–390, 2005.

[6] Craig Boutilier, Nir Friedman, Moises Goldszmidt ja Daphne Koller. Context-specific independence in Bayesian networks. Kirjassa Eric Horvitz ja Finn Verner Jensen, toim., *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, San Fransisco, CA, 1996. Morgan Kaufmann.

[7] Wray Buntine. Theory refinement on Bayesian networks. Kirjassa Bruce D'Ambrosio ja Philippe Smets, toim., *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, ss. 52–60, San Fransisco, CA, 1991. Morgan Kaufmann.

[8] David M. Chickering. Learning Bayesian networks is NP-complete. Kirjassa D. Fisher ja H.-J. Lenz, toim., *Learning from Data: AI and Statistics V*. Springer Verlag, Berliini, Saksa, 1996.

[9] David M. Chickering, David Heckerman ja Christopher Meek. A Bayesian approach to learning Bayesian networks with local structure. Kirjassa

Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence, ss. 80–89, San Francisco, CA, 1997. Morgan Kaufmann.

[10] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[11] Gregory E. Cooper ja Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[12] Selina S. Dwight, Rama Balakrishnan, Karen R. Christie, Maria C. Costanzo, Kara Dolinski, Stacia R. Engel, Becket Feierbach, Dianna G. Fisk, Jodi Hirschman, Eurie L. Hong, Laurie Issel-Tarver, Robert S. Nash an Anand Sethuraman, Barry Starr, Chandra L. Theesfeld an Rey Andrada, Gail Binkley, Qing Dong, Christopher Lane, Mark Schroeder, Shuai Weng, David Botstein ja J. Michael Cherry. Saccharomyces genome database: Underlying principles and organisation. *Briefings in Bioinformatics*, 5(1):9–22, 2004.

[13] James E. Ferrell. Self-perpetuating states in signal transduction: positive feedback, double-negative feedback and bistability. *Current Opinion in Chemical Biology*, 6:140–148, 2002.

[14] Nir Friedman ja Moises Goldszmidt. Learning Bayesian networks with local structure. Kirjassa Michael I. Jordan, toim., *Learning in Graphical Models*, ss. 421–459. MIT Press, Lontoo, Iso-Britannia, 1999.

[15] Nir Friedman, Moises Goldszmidt ja Abraham Wyner. On the application of the bootstrap for computing confidence measures on features of induced Bayesian networks. Kirjassa David Heckerman ja Joe Whittaker, toim., *Seventh International Workshop on Artificial Intelligence and Statistics*, San Francisco, CA, 1999. Morgan Kaufmann.

[16] Nir Friedman ja Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1–2):95–125, 2003.

[17] Christopher J. Fry ja Peggy J. Farnham. Context-dependent transcriptional regulation. *The Journal of Biological Chemistry*, 274(42):29583–29586, 1999.

[18] Audrey P. Gasch, Paul T. Spellman, Camilla M. Kao, Orna Carmel-Harel abd Michael B. Eisen, Gisela Storz, David Botstein ja Patrick O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11(12):4241–4257, 2000.

[19] Greg Gibson. Microarray analysis — genome-scale hypothesis scanning. *PLoS Biology*, 1(1):28–29, 2003.

[20] Alexander J. Hartemink. *Principled Computational Methods for the Validation and Discovery of Genetic Regulatory Networks.* Väitöskirja, Massachusetts Institute of Technology, 2001.

[21] Alexander J. Hartemink, David K. Gifford, Tommi S. Jaakkola ja Richard A. Young. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. Kirjassa R. Altman, A. K. Dunker, L. Hunter, K. Lauderdale ja T. Klein, toim., *Proceedings of the Sixth Pacific Symposium on Biocomputing*, ss. 422–433, Hackensack, NJ, 2001. World Scientific.

[22] David Heckerman. A tutorial on learning with Bayesian networks. Kirjassa Michael I. Jordan, toim., *Learning in Graphical Models*, ss. 301–354. MIT Press, Lontoo, Iso-Britannia, 1999.

[23] David Heckerman, Dan Geiger ja David M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[24] David Heckerman ja Eric Horvitz. Inferring informational goals from free-text queries: A Bayesian approach. Kirjassa *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, ss. 230–237, San Francisco, CA, 1998. Morgan Kaufmann.

[25] Ho-Chuan Huang ja Tsui-Ying Wang. A learning diagnosis architecture with a Bayesian network approach. Kirjassa *Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies*, ss. 33–34, 2005.

[26] Dirk Husmeier. Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics*, 19(17):2271–2282, 2003.

[27] Seiya Imoto, Takao Goto ja Satoru Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. Kirjassa *Proceedings of the Seventh Pacific Symposium on Biocomputing*, ss. 175–186, Hackensack, NJ, 2002. World Scientific Press.

[28] Christian Knüpfer, Peter Dittrich ja Clemens Beckstein. Artificial gene regulation: A data source for validation of reverse bioengineering. Kirjassa H. Schaub, F. Detje ja U. Brüggemann, toim., *Proceedings of the Sixth German Workshop on Artificial Life*, ss. 66–75, Berliini, Saksa, 2004. Akademische Verlagsgesellschacft Aka.

[29] Wai Lam ja Fahiem Bacchus. Learnings Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.

[30] Tong Ihn Lee, Nicola J. Rinaldi, François Robert, Duncan T. Odom, Ziv Bar-Joseph, Georg K. Gerber, Nancy M. Hannett, Christopher T. Harbison, Craig M. Thompson, Itamar Simon, Julia Zeitlinger, Ezra G. Jennings, Heather L. Murray, D. Benjamin Gordon, Bing Ren, John J. Wyrick, Jean-Bosco Tagne, Thomas L. Volkert, Ernest Fraenkel, David K. Gifford ja Richard A. Young. Transcriptional regulatory networks in Saccharomyces cerevisiae. *Science*, 298(5594):799–804, 2002.

[31] Bryan Lemon ja Robert Tijan. Orchestrated response: a symphony of transcription factors for gene control. *Genes & Development*, 14:2551–2569, 2000.

[32] Shoudan Liang, Stefanie Fuhrman ja Roland Somogyi. REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. Kirjassa *Proceedings of the Third Pacific Symposium on Biocomputing*, ss. 18–29, 1998.

[33] David J. Lockhart, Helin Dong, Michael C. Byrne, Maximillian T. Follettie, Michael V. Gallo, Mark S. Chee, Michael Mittmann, Chunwei Wang, Michiko Kobayashi, Heidi Norton ja Eugene L. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14:1675–1680, 1996.

[34] David Madigan ja Jeremy York. Bayesian graphical models for discrete data. *International Statistical Review*, 63(2):215–232, 1995.

[35] Agustino Martínez-Antonio ja Julio Collado-Vides. Identifying global regulators in transcriptional regulatory networks in bacteria. *Current Opinion in Microbiology*, 6:482–489, 2003.

[36] Pedro Mendes, Wei Sha ja Keying Ye. Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, 19:ii122–ii129, 2003.

[37] Judea Pearl. *Probabilistic reasoning in intelligent systems*. Morgan Kaufmann, San Mateo, CA, 1988.

[38] Dana Pe'er, Amos Tanay ja Aviv Regev. MinReg: A scalable algorithm for learning parsimonious regulatory networks in yeast and mammals. *Journal of Machine Learning Research*, 7:167–189, 2006.

[39] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[40] Jianhua Ruan ja Weixiong Zhang. A bi-dimensional regression tree approach to the modeling of gene expression regulation. *Bioinformatics*, 22(3):332–340, 2005.

[41] Mark Schena. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270(5235):467–470, 1995.

[42] Juliane Schäfer ja Korbinian Strimmer. An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–764, 2005.

[43] Eran Segal, Dana Pe'er, Aviv Regev, Daphne Koller ja Nir Friedman. Learning module networks. *Journal of Machine Learning Research*, 6:557–588, 2005.

[44] Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botsein, Daphne Koller ja Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics*, 34(2):166–176, 2003.

[45] V. Anne Smith, Erich D. Jarvis ja Alexander J. Hartemink. Evaluating functional network inference using silmulations of complex biological systems. *Bioinformatics*, 18:S216–S224, 2002.

[46] Lev A. Soinov, Maria A. Krestyaninova ja Alvis Brazma. Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biology*, 4:R6.1–R6.10, 2003.

[47] Paul T. Spellman, Gavin Sherlock, Michael Q. Zhang, Vishwanath R. Iyer, Kirk Anders, Michael B. Eisen, Patrick O. Brown, David Botstein, ja Bruce Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9:3273–3297, 1998.

[48] Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor ja Kathleen Marchal. SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7(43), 2006.

[49] Eugene P. van Someren, L. F. Wessels, E. Backer ja M. J. Reinders. Genetic network modeling. *Pharmacogenomics*, 3(4):507–525, 2002.

[50] Gong-Hong Wei, De-Pei Liu ja Chih-Chuan Liang. Charting gene regulatory networks: strategies, challenges and perspectives. *Biochemical Journal*, 381:1–12, 2004.

[51] Gregory A. Wray, Matthew W. Hahn, Ehab Abouheif, James P. Balhoff, Margaret Pizer, Matthew V. Rockman ja Laura A. Romano. The evolution of transcriptional regulation in eukaryotes. *Molecular Biology and Evolution*, 20(9):1377–1419, 2003.

[52] Jing Yu, V. Anne Smith, Paul P. Wagner, Alexander J. Hartemink ja Erich D. Jarvis. Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, 20(18):3594–3603, 2004.

[53] Daniel E. Zak, Francis J. Doyle, Gregory E. Gonye ja James S. Schwaber. Simulation studies for the identification of genetic networks from cDNA array and regulatory activity data. Kirjassa *Proceedings of the Second International Conference on Systems Biology*, ss. 231–238, 2001.

Liite A

Stressikokeiden moduulit

Seuraava taulukko esittää tiedot opituista moduuleista kokeissa, joissa toisena luokkana oli stressaavissa olosuhteissa suoritetut mittaukset ja toisen normaalitilanteen mittaukset. Jokaisesta moduulista kerrotaan siihen kuuluvat geenit, moduulia säätelevät geenit, ESR-geenien osuus kaikista moduulin geeneistä ja geeniontologialuokat, joiden Bonferroni-korjatut rikastumis-p-arvot ovat alle 0,05. 15 ensimmäistä moduulia ovat säätelijämoduuleita ja loput kohdemoduuleita.

S1	aktivoitu-ESR-osuus: 0.57	inhiboitu-ERS-osuus: 0
Geenit (N = 7): YBR026C YDR096W YDR118W YDR253C YGR023W YMR070W YPL203W		
Säätelijät: Stressille ominaiset: YPR120C Solusyklille ominaiset: YPR120C		
Rikastuneet geeniontologialuokat: –		

S2	aktivoitu-ESR-osuus: 0.38	inhiboitu-ERS-osuus: 0
Geenit (N = 13): YCR091W YDL020C YDR043C YDR277C YER020W YER054C YGL096W YGL248W YJL089W YJL141C YMR136W YOL016C YOR178C		
Säätelijät: Stressille ominaiset: YDR118W YDR096W YIL101C YLR452C Solusyklille ominaiset: YDR118W YDR096W YIL101C YLR452C		
Rikastuneet geeniontologialuokat: –		

S3	aktivoitu-ESR-osuus: 1	inhiboitu-ERS-osuus: 0
Geenit (N = 1): YPL230W		
Säätelijät: YGL180W YIR026C YMR019W		
Rikastuneet geeniontologialuokat: –		

S4	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 2): YJL098W YKR099W		
Säätelijät: YER045C YNL173C YDR096W YMR070W YDR043C		
Rikastuneet geeniontologialuokat: –		

S5	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 6): YBR135W YDL138W YER075C YMR019W YOR038C YPR120C		
Säätelijät:		
Rikastuneet geeniontologialuokat: –		

S6	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.75
Geenit (N = 4): YGL099W YGR123C YIR026C YOR101W		
Säätelijät: Stressille ominaiset: YBR026C Solusyklille ominaiset: –		
Rikastuneet geeniontologialuokat: –		

S7	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 9): YDR173C YER045C YHR136C YIL119C YJR094C YKL043W YKL109W YNL314W YPR013C		
Säätelijät: Stressille ominaiset: YLR452C Solusyklille ominaiset: –		
Rikastuneet geeniontologialuokat: –		

S8	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 2): YBR083W YGR108W		
Säätelijät: Stressille ominaiset: YLR452C YPR120C YGR023W YDR253C Solusyklille ominaiset: –		
Rikastuneet geeniontologialuokat: –		

S9	aktivoitu-ESR-osuus: 0.57	inhiboitu-ERS-osuus: 0
Geenit (N = 7): YDR085C YGL180W YIL101C YIR017C YIR018W YJL164C YNL173C		
Säätelijät: Stressille ominaiset: YPL203W YGR123C Solusyklille ominaiset: YPL203W YGR123C		
Rikastuneet geeniontologialuokat: –		

S10	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 1): YOR028C		
Säätelijät: YDR096W YDR253C YMR070W		
Rikastuneet geeniontologialuokat: –		

S11	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 6): YBL005W YFL026W YGL116W YGR249W YLR452C YOL105C		
Säätelijät: YDL170W YPL203W YGR023W		
Rikastuneet geeniontologialuokat: –		

S12	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 4): YGR109C YJL157C YPL256C YPR119W		
Säätelijät: Stressille ominaiset: YGL116W YJL141C Solusyklille ominaiset: YGL116W YJL141C		
Rikastuneet geeniontologialuokat: kinase regulator activity (p = 5.5e-05) protein kinase regulator activity (p = 5.5e-05) regulation of progression through cell cycle (p = 2.4e-03) regulation of cyclin dependent protein kinase activity (p = 9.9e-03) cyclin-dependent protein kinase regulator activity (p = 9.9e-03) enzyme regulator activity (p = 0.014) G2/M transition of mitotic cell cycle (p = 0.020) regulation of protein kinase activity (p = 0.020) regulation of transferase activity (p = 0.020)		

S13	aktivoitu-ESR-osuus: 0.33	inhiboitu-ERS-osuus: 0
Geenit (N = 3): YFL052W YJR066W YMR104C		
Säätelijät: YGL248W YIR026C YMR019W		
Rikastuneet geeniontologialuokat: –		

S14	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 1): YGL158W		
Säätelijät: YGR249W YOR028C YER075C YER045C		
Rikastuneet geeniontologialuokat: –		

S15	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 3): YDL170W YKR034W YOR032C		
Säätelijät: YIR018W YIR017C YDR253C YOR038C		
Rikastuneet geeniontologialuokat: transcription factor activity (p = 0.030)		

M16	aktivoitu-ESR-osuus: 0.20	inhiboitu-ERS-osuus: 0
Geenit (N = 59): YAL060W YBL015W YBR132C YBR183W YBR203W YCR061W YDL215C YDR178W YDR342C YDR343C YDR516C YFL016C YFL042C YFR017C YGL047W YGL104C YGL259W YGR130C YGR194C YGR205W YGR243W YGR250C YGR289C YHL035C YHR016C YIL055C YIL065C YIL113W YIL155C YJL082W YJL213W YJL217W YKL035W YKL036C YKL085W YKL187C YKR058W YKR098C YLR177W YLR259C YLR338W YLR348C YLR438W YML070W YML091C YML120C YMR291W YNL077W YNL305C YNR014W YOL155C YOR136W YOR137C YOR317W YOR347C YOR386W YPL247C YPR154W YPR160W		
Säätelijät: Stressille ominaiset: YKR099W YMR136W YPL230W YJL164C Solusyklille ominaiset: YKR099W YMR136W YPL230W YJL164C		
Rikastuneet geeniontologialuokat: –		

M17	aktivoitu-ESR-osuus: 0.7	inhiboitu-ERS-osuus: 0
Geenit (N = 23): YCL035C YEL012W YER142C YGR019W YHL024W YHR029C YIL111W YIR016W YIR039C YJR096W YKL065C YMR110C YMR181C YNL015W YNL115C YOL032W YOL071W YOL082W YOL083W YOR121C YOR285W YPL004C YPL165C		
Säätelijät: Stressille ominaiset: YGL248W YGR123C YGL096W YPL230W Solusyklille ominaiset: YGL248W YGR123C YGL096W YPL230W		
Rikastuneet geeniontologialuokat: –		

M18	aktivoitu-ESR-osuus: 1	inhiboitu-ERS-osuus: 0
Geenit (N = 34): YBR056W YCL040W YCL042W YDL021W YDL204W YDR074W YDR171W YEL011W YER053C YER079W YER150W YFL014W YGR008C YGR043C YGR088W YGR248W YHL021C YKL091C YKL103C YKL142W YKL150W YLL026W YLR149C YLR178C YLR258W YLR312C YLR327C YML100W YML128C YMR105C YMR250W YNL160W YNL274C YOR052C		
Säätelijät: Stressille ominaiset: YPL230W YLR452C YGR249W YDR085C Solusyklille ominaiset: YPL230W YLR452C YDR085C		
Rikastuneet geeniontologialuokat: –		

M19	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.14
Geenit (N = 57): YAL012W YAL023C YAL035W YAR010C YBL012C YBR068C YBR206W YCL025C YCL027W YDL055C YDL084W YDL093W YDR046C YDR266C YDR309C YDR432W YDR433W YDR461W YDR508C YDR509W YDR518W YFL037W YGL012W YGL021W YGL032C YGL077C YGL202W YGR055W YGR124W YGR138C YGR172C YGR185C YGR279C YIL015W YJL107C YJL108C YJL158C YJL170C YKL209C YKR042W YLR060W YLR153C YLR300W YLR378C YML126C YMR032W YMR215W YNL058C YNL145W YNR043W YNR061C YOL020W YOR198C YOR247W YPL014W YPL028W YPR074C		
Säätelijät: YDR253C YGL099W YER045C YPR120C YIR026C		
Rikastuneet geeniontologialuokat: –		

M20	aktivoitu-ESR-osuus: 0.16	inhiboitu-ERS-osuus: 0
Geenit (N = 19): YBR244W YDR032C YDR132C YFL057C YHR008C YJL101C YKL071W YKR066C YLL060C YLR108C YLR109W YLR152C YLR303W YLR460C YML116W YMR318C YNL331C YOL165C YOR382W		
Säätelijät: Stressille ominaiset: YPR013C YGL096W YJL141C YKR099W Solusyklille ominaiset: YPR013C		
Rikastuneet geeniontologialuokat: –		

M21	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.94

Geenit (N = 81):

YBL039C YBL068W YBR069C YBR142W YBR238C YCL037C YCL054W YCR057C YCR072C YDL148C YDR060W YDR101C YDR324C YDR365C YDR398W YDR496C YER006W YER110C YGL078C YGL111W YGL120C YGL171W YGR103W YGR128C YGR145W YGR160W YGR245C YGR280C YHR170W YHR196W YHR197W YJL033W YJL109C YJR041C YJR070C YKL078W YKR092C YLL008W YLL011W YLR175W YLR196W YLR197W YLR198C YLR222C YLR249W YLR276C YLR401C YLR409C YLR413W YLR449W YML093W YMR049C YMR093W YMR128W YMR131C YMR229C YMR290C YNL061W YNL132W YNL174W YNL175C YNL248C YNL313C YOL041C YOL077C YOR095C YOR206W YOR243C YOR272W YOR309C YOR310C YPL043W YPL044C YPL093W YPL126W YPL183C YPL211W YPL226W YPR110C YPR112C YPR190C

Säätelijät:

Stressille ominaiset: YPL230W YOR032C YKR099W YGL099W

Solusyklille ominaiset: YPL230W YOR032C YKR099W YGL099W

Rikastuneet geeniontologialuokat:

nucleolus (p = 8e-31)
ribosome biogenesis (p = 4e-26)
ribosome biogenesis and assembly (p = 7e-26)
cytoplasm organization and biogenesis (p = 7e-26)
rRNA metabolism (p = 9.3e-21)
organelle organization and biogenesis (p = 8.3e-20)
rRNA processing (p = 4.4e-19)
nucleus (p = 1.5e-18)
RNA processing (p = 3.1e-18)
RNA metabolism (p = 1.9e-17)
cell organization and biogenesis (p = 7.1e-17)
non-membrane-bound organelle (p = 1.2e-14)
intracellular non-membrane-bound organelle (p = 1.2e-14)
nucleobase, nucleoside, nucleotide and nucleic acid metabolism (p = 7.3e-13)
RNA binding (p = 1.3e-08)
snoRNA binding (p = 5.4e-08)
processing of 20S pre-rRNA (p = 2.5e-07)
biopolymer metabolism (p = 2.7e-07)
small nucleolar ribonucleoprotein complex (p = 6.8e-07)
membrane-bound organelle (p = 8.3e-07)
intracellular membrane-bound organelle (p = 8.3e-07)
nucleic acid binding (p = 1.2e-04)
35S primary transcript processing (p = 2.0e-04)
organelle (p = 1.0e-03)
intracellular organelle (p = 1.0e-03)
RNA helicase activity (p = 6e-03)
helicase activity (p = 0.019)
binding (p = 0.027)

M22	aktivoitu-ESR-osuus: 0.27	inhiboitu-ERS-osuus: 0

Geenit (N = 15):

YBR008C YGL117W YGR111W YGR209C YHL036W YJR059W YKL201C YKL202W YKR067W YLR216C YMR103C YMR184W YNL055C YOR020C YPR167C

Säätelijät:

Stressille ominaiset: YOL016C YKR099W

Solusyklille ominaiset: YOL016C

Rikastuneet geeniontologialuokat:

–

M23	aktivoitu-ESR-osuus: 0.029	inhiboitu-ERS-osuus: 0
Geenit (N = 35): YBR105C YBR294W YCL018W YDL238C YEL063C YER175C YGL059W YGL184C YGR121C YGR154C YIL165C YIR027C YIR028W YIR029W YIR030C YIR031C YIR032C YIR042C YJR152W YKR033C YKR039W YLR092W YLR155C YLR157C YLR158C YLR160C YLR174W YMR096W YNL117W YNL276C YNL277W YNR064C YOL058W YOL163W YOR303W		
Säätelijät: YDL170W YIL101C YGL116W YIR018W		
Rikastuneet geeniontologialuokat: hydrolase activity, acting on carbon-nitrogen (but not peptide) bonds (p = 2.5e-04) nitrogen compound metabolism (p = 3.5e-04) amine metabolism (p = 1.1e-03) cellular response to nitrogen starvation (p = 0.041)		

M24	aktivoitu-ESR-osuus: 0.95	inhiboitu-ERS-osuus: 0
Geenit (N = 19): YBL064C YBR169C YDL124W YDR453C YDR513W YDR533C YIR038C YKR076W YLR270W YML004C YML131W YMR090W YMR173W YMR315W YNL134C YOL150C YOL151W YOR120W YOR289W		
Säätelijät: Stressille ominaiset: YDR277C YHR136C YPL230W YDR085C Solusyklille ominaiset: YDR277C YHR136C YPL230W YDR085C		
Rikastuneet geeniontologialuokat: –		

M25	aktivoitu-ESR-osuus: 0.023	inhiboitu-ERS-osuus: 0
Geenit (N = 43): YBR179C YBR284W YDL059C YDL200C YDR022C YDR041W YDR159W YDR202C YDR219C YDR262W YDR425W YDR484W YEL072W YER024W YER033C YER046W YGL041C YGL177W YGR053C YHL016C YHR114W YIL024C YIL164C YIR034C YJL015C YKL218C YKR006C YKR069W YKR097W YLL055W YLR053C YLR168C YMR114C YMR133W YMR252C YNL193W YNL260C YOR003W YOR389W YPL024W YPL147W YPR012W YPR194C		
Säätelijät: Stressille ominaiset: YDR118W YIR018W YER020W Solusyklille ominaiset: YDR118W YIR018W YER020W		
Rikastuneet geeniontologialuokat: –		

M26	aktivoitu-ESR-osuus: 0.19	inhiboitu-ERS-osuus: 0
Geenit (N = 26): YAL005C YBL075C YBR054W YCR005C YCR021C YDL222C YDR055W YDR214W YDR380W YER103W YFR015C YGR052W YGR142W YHR137W YHR140W YJL034W YKL096W YLR217W YNL007C YOR027W YOR273C YOR383C YPL061W YPL240C YPL265W YPR158W		
Säätelijät: YKR099W YIL101C YKR034W YMR019W YLR452C		
Rikastuneet geeniontologialuokat: protein folding (p = 2.3e-03)		

M27	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.8
Geenit (N = 40): YBL054W YBR266C YBR267W YCL059C YDL063C YDR234W YDR312W YDR399W YDR492W YER056C YGL029W YGR187C YHR088W YHR169W YIL019W YJL069C YJL122W YJL148W YJL200C YJR002W YKL009W YKL029C YKL082C YKL172W YKR024C YKR081C YLR221C YML043C YMR014W YMR239C YNL110C YNL112W YNL141W YNL299W YOR004W YOR056C YOR287C YOR294W YPL146C YPR142C		
Säätelijät: Stressille ominaiset: YBL005W YDL170W YKR099W YGL099W Solusyklille ominaiset: YBL005W YDL170W YKR099W		
Rikastuneet geeniontologialuokat: nucleolus (p = 2.2e-06) ribosome biogenesis and assembly (p = 4.6e-06) cytoplasm organization and biogenesis (p = 4.6e-06) rRNA metabolism (p = 1e-05) ribosome biogenesis (p = 1.8e-05) rRNA processing (p = 5.5e-05) nucleus (p = 6.7e-05) RNA processing (p = 3.2e-04) RNA metabolism (p = 4.4e-04) organelle organization and biogenesis (p = 6e-04) cell organization and biogenesis (p = 7.2e-03) nucleobase, nucleoside, nucleotide and nucleic acid metabolism (p = 7.9e-03) membrane-bound organelle (p = 0.045) intracellular membrane-bound organelle (p = 0.045)		

M28	aktivoitu-ESR-osuus: 0.027	inhiboitu-ERS-osuus: 0
Geenit (N = 74): YAL054C YBL049W YBR077C YBR099C YCL048W YDL085W YDL218W YDR030C YDR275W YDR313C YDR424C YDR473C YDR485C YDR536W YER039C YER065C YER096W YFL053W YGR087C YGR153W YGR225W YGR230W YGR236C YHR006W YHR053C YHR055C YHR139C YIL167W YJL153C YJR011C YJR019C YJR078W YJR095W YKL050C YKL107W YKL221W YLL056C YLR004C YLR030W YLR031W YLR054C YLR128W YLR254C YLR311C YLR346C YLR377C YMR077C YMR118C YMR322C YNL006W YNL009W YNL202W YNL203C YNL335W YNR068C YNR069C YOL050C YOR031W YOR049C YOR186W YOR348C YOR392W YOR394W YPL018W YPL088W YPL095C YPL107W YPL119C YPL185W YPL222W YPL280W YPR007C YPR015C YPR193C		
Säätelijät: YDR085C YIR017C YKR099W YPR013C YDR118W		
Rikastuneet geeniontologialuokat: cellular component unknown (p = 6e-03)		

M29	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.12
Geenit (N = 17): YAR071W YBL003C YBR067C YBR089W YCL064C YDR225W YDR384C YGL055W YGL255W YGR234W YHR215W YKR013W YLR183C YMR305C YOR313C YOR315W YPL163C		
Säätelijät: YOR038C YGR023W YGR123C YPL256C YKR034W		
Rikastuneet geeniontologialuokat: –		

M30	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.66

Geenit (N = 74):

YBL076C YBR031W YBR079C YBR162C YBR249C YCR034W YCR059C YDR012W YDR037W YDR091C YDR144C YDR429C YDR507C YEL040W YEL042W YER025W YER036C YER043C YER049W YER055C YER156C YFL002C YGL008C YGL014W YGL148W YGR061C YGR094W YGR159C YGR162W YGR173W YGR177C YGR200C YGR285C YHL011C YHL012W YHL015W YHR019C YHR025W YHR094C YIL078W YIL158W YJL080C YJR063W YJR143C YKL004W YKR059W YLR083C YLR146C YLR180W YLR340W YLR372W YMR246W YMR307W YMR308C YMR309C YNL087W YNL247W YNL292W YNR038W YOL092W YOL097C YOR051C YOR207C YOR248W YOR361C YPL086C YPL131W YPL160W YPL237W YPL238C YPL263C YPR033C YPR145W YPR163C

Säätelijät:

Stressille ominaiset: YGR123C YPL203W YJL157C YJL164C

Solusyklille ominaiset: –

Rikastuneet geeniontologialuokat:

translation (p = 1.2e-07)
translation initiation factor activity (p = 2.0e-05)
translational initiation (p = 5e-05)
cellular biosynthesis (p = 6.6e-05)
tRNA ligase activity (p = 2.4e-04)
RNA ligase activity (p = 2.4e-04)
ligase activity, forming carbon-oxygen bonds (p = 2.4e-04)
ligase activity, forming aminoacyl-tRNA and related compounds (p = 2.4e-04)
biosynthesis (p = 3.4e-04)
ligase activity, forming phosphoric ester bonds (p = 6.4e-04)
protein biosynthesis (p = 2.1e-03)
translation factor activity, nucleic acid binding (p = 3.3e-03)
translation regulator activity (p = 6.8e-03)
macromolecule biosynthesis (p = 0.017)
ligase activity (p = 0.022)
cellular protein metabolism (p = 0.024)
tRNA aminoacylation for protein translation (p = 4e-02)
amino acid activation (p = 4e-02)
tRNA aminoacylation (p = 4e-02)

M31	aktivoitu-ESR-osuus: 0.47	inhiboitu-ERS-osuus: 0

Geenit (N = 75):

YAL034C YAL061W YBR018C YBR046C YBR116C YBR117C YBR241C YBR280C YBR285W YDL024C YDL169C YDL223C YDR018C YDR031W YDR058C YDR204W YDR358W YEL041W YER035W YER121W YER158C YGL081W YGL250W YGR066C YGR201C YIL045W YIL097W YIL099W YIL107C YJL045W YJL057C YJL066C YJL116C YJL142C YJL155C YJL163C YJR008W YJR155W YKL026C YKL093W YKL148C YKL188C YKR009C YKR046C YLL020C YLL041C YLR164W YLR267W YLR356W YML042W YML054C YML118W YMR053C YMR068W YMR081C YMR107W YMR271C YNL093W YNL195C YNL237W YNR002C YOL047C YOL048C YOL152W YOR019W YOR036W YOR134W YOR391C YPL123C YPL166W YPL223C YPR026W YPR030W YPR150W YPR151C

Säätelijät:

YIL101C YGL180W YDR085C YCR091W YJL164C

Rikastuneet geeniontologialuokat:

cellular component unknown (p = 0.037)

M32	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.96

Geenit (N = 92):

YAR073W YBL087C YBR048W YBR181C YBR189W YBR191W YCR031C YDL014W YDL051W YDL061C YDL081C YDL082W YDL083C YDR025W YDR321W YDR385W YDR417C YDR418W YDR450W YEL021W YEL026W YEL054C YER074W YER102W YER117W YGL030W YGL031C YGL076C YGL123W YGL135W YGL147C YGR034W YGR085C YGR148C YGR214W YGR264C YHL033C YHR010W YHR020W YHR203C YHR216W YIL018W YIL133C YJL138C YJL177W YJL190C YJR071W YJR123W YJR145C YKL006W YKL056C YKL081W YKL180W YKL181W YKL216W YLL044W YLL045C YLR029C YLR048W YLR061W YLR062C YLR325C YLR339C YLR367W YLR388W YLR432W YLR441C YLR448W YML022W YMR321C YNL067W YNL069C YNL119W YNL178W YNL209W YNL301C YOL040C YOL120C YOR063W YOR096W YOR133W YOR167C YOR224C YOR369C YPL079W YPL090C YPL142C YPL220W YPL273W YPR010C YPR044C YPR102C

Säätelijät:

YGL248W YGR123C YNL173C YJL157C YGL099W

Rikastuneet geeniontologialuokat:

ribosome (p = 1.1e-45)
structural constituent of ribosome (p = 1.8e-45)
cytosolic ribosome (sensu Eukaryota) (p = 1.9e-44)
protein biosynthesis (p = 2.7e-43)
structural molecule activity (p = 1.1e-41)
macromolecule biosynthesis (p = 6e-40)
ribonucleoprotein complex (p = 9.9e-38)
cellular biosynthesis (p = 2.6e-37)
cytosol (p = 1.5e-35)
biosynthesis (p = 2.5e-35)
cellular protein metabolism (p = 4.3e-30)
protein complex (p = 3.2e-29)
protein metabolism (p = 9.1e-29)
non-membrane-bound organelle (p = 2.8e-25)
intracellular non-membrane-bound organelle (p = 2.8e-25)
cellular macromolecule metabolism (p = 1.6e-23)
cytosolic large ribosomal subunit (sensu Eukaryota) (p = 1.7e-21)
large ribosomal subunit (p = 3.7e-21)
macromolecule metabolism (p = 3.9e-21)
cytosolic small ribosomal subunit (sensu Eukaryota) (p = 2.1e-14)
eukaryotic 48S initiation complex (p = 2.1e-14)
small ribosomal subunit (p = 4.4e-14)
primary metabolism (p = 1.1e-13)
eukaryotic 43S preinitiation complex (p = 1.2e-12)
cellular metabolism (p = 5.3e-11)
metabolism (p = 6.3e-10)
cytoplasm (p = 8.5e-05)
cellular physiological process (p = 5e-02)

M33	aktivoitu-ESR-osuus: 0.37	inhiboitu-ERS-osuus: 0

Geenit (N = 27):

YDL022W YDL023C YDR077W YDR209C YDR391C YER062C YIL117C YJR073C YKR011C YKR091W YLR120C YLR121C YLR194C YLR350W YLR414C YML130C YMR007W YMR040W YMR251W YMR316W YNL208W YNL241C YOR185C YOR220W YOR385W YPL087W YPL154C

Säätelijät:

Stressille ominaiset: YIR018W YOL016C YDR085C YIR026C
Solusyklille ominaiset: YIR018W YOL016C YDR085C YIR026C

Rikastuneet geeniontologialuokat:

–

M34	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 37): YAL044C YBL005W-A YBL101W-A YBR012W-B YCL030C YDR019C YEL071W YER091C YER138C YER160C YGL009C YGL062W YJL060W YJL172W YJR010W YJR026W YJR027W YJR028W YJR029W YJR109C YJR137C YLL062C YLR058C YLR304C YML040W YML045W YMR046C YMR051C YMR189W YNL037C YNR050C YOL064C YOL140W YOR128C YOR135C YOR375C YPL250C		
Säätelijät: Stressille ominaiset: YGL248W YBL005W YDR253C YJR094C Solusyklille ominaiset: YGL248W YDR253C YJR094C		
Rikastuneet geeniontologialuokat: amino acid metabolism (p = 2.3e-06) carboxylic acid metabolism (p = 2.7e-06) organic acid metabolism (p = 2.7e-06) amino acid and derivative metabolism (p = 5.3e-06) nitrogen compound metabolism (p = 7e-06) amine metabolism (p = 2.1e-05) amino acid biosynthesis (p = 4e-05) nitrogen compound biosynthesis (p = 5.8e-05) amine biosynthesis (p = 5.8e-05)		

M35	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0
Geenit (N = 9): YBR088C YBR158W YER124C YGL028C YHR143W YLR286C YNL078W YNL327W YNR044W		
Säätelijät: Stressille ominaiset: YFL026W YER045C Solusyklille ominaiset: YFL026W YER045C YPL256C YBR083W		
Rikastuneet geeniontologialuokat: cell separation during cytokinesis (p = 0.021) cytokinesis, completion of separation (p = 0.021)		

M36	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.17
Geenit (N = 35): YBR104W YBR271W YCL036W YCR063W YDL122W YDR524C YEL044W YER073W YGR030C YGR079W YGR081C YGR211W YGR251W YHR070W YHR072W YHR204W YIL079C YJL212C YJR132W YJR150C YKL044W YLR056W YLR405W YML018C YML123C YMR138W YMR185W YMR319C YNL023C YNL065W YOR048C YOR295W YOR342C YOR359W YPR187W		
Säätelijät: Stressille ominaiset: YKR099W YMR136W YGL099W Solusyklille ominaiset: –		
Rikastuneet geeniontologialuokat: –		

M37	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.82

Geenit (N = 90):

YAL025C YBL028C YBR247C YCR016W YCR087W YDL031W YDL050C YDL060W YDL062W YDL150W YDL152W YDR083W YDR087C YDR120C YDR165W YDR184C YDR361C YDR412W YDR413C YDR449C YDR465C YDR491C YER082C YER126C YER127W YGL169W YGL170C YGR283C YHR052W YHR065C YHR148W YIL064W YIL096C YIL127C YJL198W YJL208C YJR003C YJR097W YKL021C YKL099C YKL156W YKL191W YKR056W YKR060W YKR079C YLL035W YLR002C YLR003C YLR009W YLR063W YLR073C YLR074C YLR186W YLR397C YLR400W YLR435W YML080W YMR310C YNL022C YNL062C YNL075W YNL113W YNL151C YNL255C YNL308C YNR003C YNR012W YNR053C YNR054C YOL010W YOL022C YOL079W YOL124C YOL125W YOL141W YOL144W YOR001W YOR021C YOR078W YOR091W YOR145C YOR146W YOR252W YOR340C YPL207W YPL212C YPL266W YPR137W YPR143W YPR144C

Säätelijät:

Stressille ominaiset: YOL016C YGL099W YKR099W YOR028C

Solusyklille ominaiset: –

Rikastuneet geeniontologialuokat:

ribosome biogenesis (p = 1.8e-10)
RNA metabolism (p = 1.6e-09)
nucleolus (p = 2.5e-09)
nucleus (p = 3.0e-09)
ribosome biogenesis and assembly (p = 5.8e-08)
cytoplasm organization and biogenesis (p = 5.8e-08)
rRNA processing (p = 5.1e-07)
RNA processing (p = 1.4e-06)
rRNA metabolism (p = 1.5e-06)
nucleobase, nucleoside, nucleotide and nucleic acid metabolism (p = 7.8e-06)
biopolymer metabolism (p = 1.1e-04)
organelle organization and biogenesis (p = 1.5e-04)
cell organization and biogenesis (p = 6.3e-03)
35S primary transcript processing (p = 8.9e-03)

M38	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0

Geenit (N = 122):

YAL045C YAL046C YAL067C YAR042W YBL026W YBL033C YBL108W YBR097W YBR129C YBR138C YBR168W YBR180W YBR192W YBR208C YBR261C YBR299W YCL016C YCL068C YCR018C YCR037C YCR085W YCR101C YDL001W YDL030W YDL065C YDL246C YDR130C YDR156W YDR180W YDR457W YDR520C YEL051W YEL057C YEL062W YEL064C YER011W YER015W YER071C YER104W YER181C YER183C YFL040W YFR026C YFR032C YGL088W YGL138C YGL141W YGL223C YGL235W YGR020C YGR059W YGR075C YGR082W YGR184C YGR291C YHL031C YHR014W YHR033W YHR107C YHR154W YHR185C YIL166C YIR019C YJL027C YJL120W YJL156C YJL162C YJL168C YJL216C YJR111C YKL052C YKL083W YKL097C YKL199C YKL223W YKR005C YKR073C YKR103W YLL063C YLR012C YLR087C YLR114C YLR235C YLR255C YLR296W YLR349W YLR416C YML032C YML061C YML107C YMR074C YMR122C YMR162C YMR219W YNL034W YNL108C YNL198C YNL221C YOL056W YOL066C YOL085C YOL091W YOR016C YOR029W YOR037W YOR177C YOR393W YPL008W YPL017C YPL023C YPL025C YPL042C YPL062W YPL144W YPL148C YPL167C YPL181W YPL200W YPL257W YPR135W YPR182W YPR189W

Säätelijät:

Stressille ominaiset: YGL099W YDL170W YGL116W

Solusyklille ominaiset: YGL099W YDL170W YCR091W

Rikastuneet geeniontologialuokat:

–

M39	aktivoitu-ESR-osuus: 0.88	inhiboitu-ERS-osuus: 0

Geenit (N = 49):

YBL078C YBR006W YBR072W YBR126C YBR230C YCR062W YDL110C YDR001C YDR070C YDR258C YEL039C YEL060C YER037W YER119C YGL121C YGL156W YGR161C YGR244C YGR256W YHR096C YHR097C YHR112C YHR138C YHR209W YIL136W YJL048C YJL088W YJL144W YJL161W YKL151C YLL039C YLR080W YLR142W YLR205C YLR251W YLR252W YLR299W YMR195W YMR196W YNL036W YNL200C YOL084W YOL153C YOR161C YOR173W YOR374W YPL054W YPL186C YPR184W

Säätelijät:

Stressille ominaiset: YGL096W YGR123C YPL230W YIL101C

Solusyklille ominaiset: YGL096W YGR123C YPL230W YIL101C

Rikastuneet geeniontologialuokat:

–

M40	aktivoitu-ESR-osuus: 0	inhiboitu-ERS-osuus: 0.96

Geenit (N = 47):

YAL003W YBL027W YBL092W YBR084C-A YDR064W YDR382W YDR447C YDR471W YDR500C YER131W YGL102C YGL103W YGR027C YGR118W YHL001W YHR021C YHR141C YIL052C YJL136C YJL188C YJL189W YJL191W YKR057W YKR094C YLR075W YLR076C YLR150W YLR167W YLR185W YLR264W YLR333C YLR344W YML063W YML106W YOL039W YOL121C YOL127W YOR234C YOR293W YOR312C YPL081W YPL143W YPL189W YPL197C YPL198W YPR043W YPR132W

Säätelijät:

YGL248W YJL089W YGR123C YJR094C YIR018W

Rikastuneet geeniontologialuokat:

cytosolic ribosome (sensu Eukaryota) (p = 1.6e-33)
ribosome (p = 1.8e-32)
structural constituent of ribosome (p = 3.4e-32)
structural molecule activity (p = 4.5e-30)
protein biosynthesis (p = 8.2e-27)
cytosol (p = 1.5e-26)
ribonucleoprotein complex (p = 1.7e-25)
macromolecule biosynthesis (p = 5e-25)
cellular biosynthesis (p = 1.0e-20)
biosynthesis (p = 1.1e-19)
protein complex (p = 1.7e-18)
cellular protein metabolism (p = 3.2e-18)
protein metabolism (p = 1.7e-17)
non-membrane-bound organelle (p = 6e-17)
intracellular non-membrane-bound organelle (p = 6e-17)
cellular macromolecule metabolism (p = 1.3e-14)
cytosolic large ribosomal subunit (sensu Eukaryota) (p = 2.1e-13)
macromolecule metabolism (p = 3.0e-13)
large ribosomal subunit (p = 3.2e-13)
cytosolic small ribosomal subunit (sensu Eukaryota) (p = 2.2e-09)
eukaryotic 48S initiation complex (p = 2.2e-09)
small ribosomal subunit (p = 3.3e-09)
eukaryotic 43S preinitiation complex (p = 2.3e-08)
primary metabolism (p = 4.3e-04)
cellular metabolism (p = 8e-03)
cytoplasm (p = 0.017)
metabolism (p = 0.026)