KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kasteelpark Arenberg 10, 3001 Leuven (Heverlee)

ROBUST ALGORITHMS FOR INFERRING REGULATORY NETWORKS BASED ON GENE EXPRESSION MEASUREMENTS AND BIOLOGICAL PRIOR INFORMATION

Promotoren:
Prof. dr. ir. B. De Moor
Prof. dr. ir. K. Marchal

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de ingenieurswetenschappen

door

Tim VAN DEN BULCKE

Mei 2009

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kasteelpark Arenberg 10, 3001 Leuven (Heverlee)

ROBUST ALGORITHMS FOR INFERRING REGULATORY NETWORKS BASED ON GENE EXPRESSION MEASUREMENTS AND BIOLOGICAL PRIOR INFORMATION

Jury:
Prof. dr. ir. J. Berlamont, voorzitter
Prof. dr. ir. B. De Moor, promotor
Prof. dr. ir. K. Marchal, promotor
Prof. dr. ir. J.A.K. Suykens
Prof. dr. L. De Raedt
Prof. dr. G. Verbeke
Prof. dr. ir. T. De Bie (University of Bristol, U.K.)
Dr. T. Michoel (Universiteit Gent)

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de ingenieurswetenschappen

door

Tim VAN DEN BULCKE

U.D.C. 681.3*J3 Mei 2009

ISBN 978-94-6018-068-2

U.D.C. 681.3*J3

D/2009/7515/51

Dankwoord

Het lijkt wel gisteren dat ik op ESAT begonnen ben. Die 'eerste dag' was meteen ook de eerste dag van de grote vakantie met prachtig weer en ik kwam bijgevolg aan op een nagenoeg lege derde verdieping. Kathleen, de efficiëntie zelve, zei toen *'Okee, begin maar hé!'* en daarmee was het startschot gegeven. U ziet, de start van een doctoraat hoeft niet altijd moeilijk te zijn, dat komt vanzelf wel nadien.

Op elke publicatie staat steeds een uitgebreide auteurslijst maar vreemd genoeg staat op een doctoraat slechts één naam (geachte promotoren en juryleden, mijn excuses voor deze dichterlijke vrijheid). Nochthans is een doctoraat nooit mogelijk zonder de intense samenwerking van een heleboel mensen.

Allereerst wens ik daarom mijn promotoren prof. Bart De Moor en prof. Kathleen Marchal te bedanken, het is in de eerste plaats dankzij hen dat ik hier de afgelopen jaren heb mogen en kunnen werken.

Bart, dank voor je niet aflatende steun zowel voor als achter de schermen en dit zowel binnen als buiten mijn doctoraat. Als ik iets meedraag van ESAT dan is het wel dat hier alles mogelijk is als je er maar hard genoeg voor werkt. Kathleen, jouw niet aflatend enthousiasme en werkkracht zijn steeds een bron van inspiratie geweest. Het is dan ook met heel veel respect en bewondering dat ik de afgelopen jaren met je heb mogen samenwerken. Op 'den chat' was er altijd tijd om een probleem aan te kaarten maar ook om de laatste nieuwtjes uit te wisselen. Er bleek zelfs een heuse 24/7 permanentie te zijn, want wanneer ik ook inlogde, jij was altijd aan het werk. Bedankt om me zoveel richting en tegelijkertijd zoveel vrijheid te geven, ik had het niet beter kunnen wensen!

Hui and Lore, I can safely say that I couldn't have done this without you. *'Thank you!'* by far doesn't cover how I feel and neither do the chocolates cover for all the help you gave. Hui, half a word was always enough to understand the problem. Strange enough, an *'Ok, I get it, next.'* was often heard halfway an explanation, an indication that you understood things better than I did. Since I know how much you appreciate *Vienna-by-night* tours, we'll definitely do that again someday. I wish you all the best with your baby-to-come!

Koen, één van de fijnste periodes van mijn doctoraat heeft zich ongetwijfeld

'in Antwerpen' afgespeeld. De beste koffie - het spijt me, dit wordt even pijnlijk voor sommige Leuvense lezers - werd in Antwerpen geserveerd, inclusief koekjes. Koen, je enthousiasme werkte ongelooflijk aanstekelijk en je scherpe inzicht was vaak verbluffend. Van newbie heb je je opgewerkt tot een echte linux-guru en voor jou was het motto vaak: hoe moeilijker de opdracht, hoe plezanter de uitdaging. Ons gezamelijk project heeft geleid tot een hechte vriendschap en vele fijne momenten, de nachtelijke opsluiting in het gebouw reken ik daar gemakshalve ook bij. Bart en Piet, ik heb de no-nonsens mentaliteit op ISLab steeds geapprecieerd en jullie nuchtere kijk op ons werk leverde vaak de nodige brandstof om de juiste experimenten te doen. Kris, Kim en Hai, onze paden hebben elkaar op ISLab slechts kort gekruist maar de fijne babbels op congres hebben op dat vlak veel goedgemaakt.

Olivier, Thomas en Peter, de koffie is nooit het belangrijkste element geweest van onze koffiepauzes denk ik. Veel ideeën, goede inzichten en wereldreddende filosofieën hebben hier hun oorsprong gevonden en al even vaak ook hun einde. Olivier, alle interacties zullen spijtig genoeg virtueel worden het komende jaar, ik wens je veel succes in Stanford! Wout en Thomas, de verhalen over onze thesisstudenten zullen nog wel een paar kampvuren meegaan. Het was fijn om een eiland te mogen delen met jullie! Er werd hard gewerkt, maar er was altijd tijd voor hulp bij kleine en grote problemen. Daarnaast hebben we er ook veel *leute* gehad. De 'pop', de 'shrimp-in-a-tube' en de kartonnen torens verschenen altijd als het net iets té stil werd. Sonia, Jiqiu and Sylvain, you are the fine new inhabitants of our little island, protect it at all costs against *'the others'*! It was and still is a pleasure to have such nice and clever colleagues right next to me.

Nothing ever stayed the same at the third floor and I had the pleasure of sharing office with so many people over the years. Kristof, Ruth, Pieter, Anneleen, Ernesto, Lieven, Peter, Shi, Tunde, Daniela, Leon, Joke, Cynthia, Karen, Liesbeth, Olivier, Frank, Gert, Yves, Thomas, Wout, Sonia, Jiqiu and Sylvain, you each gave your own personal touch to our floor. Thanks for all the nice shared moments! Dear colleagues in Ghent and BioFrame members, it was a pleasure working with all of you and I hope we can further collaborate (and ski!) in the future.

Onder het motto *'Goed werk wordt nooit geleverd op een lege maag.'* waren we jaren vaste klant in de *ViaVia*. Naast de harde kern Karen, Raf, Olivier, Peter en Thomas waren er regelmatig ook 'invited guests' zoals Francesca, al is de befaamde Via-spaghetti bij haar nooit echt in de smaak gevallen. De game evenings zijn ondertussen ook een vast fenomeen geworden en leren *Weerwolven* staat nu geloof ik vast in het programma van eerstejaars doctoraatsstudenten.

Op de vierde verdieping zijn de meest luidruchtige collega's wijselijk op twee aparte eilandjes geplaatst achter een lange reeks kasten. Tom, Bert, Raf, Niels, Nathalie, Frizo, Bert and Steven, het was altijd plezant om eens langs te komen!

Daar waren ook 'onze' IT mensen gestationeerd en zij speelden steeds snel in op elk nieuw probleem: Edwin, Maarten en Kris, bedankt! Ida, Ilse en Mimi jullie regelden achter de schermen voor ons alle zaken. Bedankt daarvoor en voor het geduld telkens er een documentje te weinig of teveel was ingevuld.

While being stationed at ESAT, I had the fortune of actually be part of two bioinformatics groups. My fine colleagues at agriculture, I couldn't have wished for nicer colleagues than you. The wonderful international mix was always food for passionate conversations and yet we bind together as such a strong team. Kathleen, Kristof, Inge, Sigrid, Carolina, Valerie, Marleen, Fu, Alejandro, Lyn, Abeer, Riet, Lore, Karen, Hui, Sunny, Peyman, Ivan, Jo, Pieter and *los Cubanos* Aminael and Roldan!

Kristof, the sometimes animated microarray normalization discussions resulted in both great insights and usually even more questions, especially those at 5am during our brainstorms. Aminael, it is amazing how quickly we connected in Vienna and how much common interests we shared there! Peyman, I am still curious about your 'updated' Werewolves strategy and Fu, your energy is as amazing as the speed at which you can fall asleep after lunch. Thanks to the Arabian nights that Abeer organized, I will always remember Arabian coffee as something very special. And Valerie, how you combine such a warm personality with your passion for work is something I can only admire. Carolina, I don't think I ever saw you *not* smiling! Thank you all for the cold skiing trips, the warm friendships and the sleep-depriving brainstorms! Working at agriculture has always felt like being in one big international family where 'mama' Kathleen took great care of her youngsters.

Mams en paps, hoe kan ik in enkele zinnetjes samenvatten wat jullie al een leven lang doen voor me? Jullie hebben me alle kansen gegeven om me te ontplooien. Jullie stonden altijd klaar, soms aan de zijlijn en soms vooraan, om bij te staan met raad en daad waar nodig en tegelijk ook zonder dwang, zodat we elk zelf op ontdekkingsreis konden gaan in het leven. Zonder de intellectuele bagage van thuis en '*Zijt gij nu weeral beneden?*' tijdens de examentijd, zou dit doctoraat er nooit geweest zijn. Bedankt voor alles! Stijn en Bram, we hebben elkaar in de drukte van de laatste tijd wat minder gezien en gehoord, maar we gaan de schade terug inhalen nu!

Dank ook aan de juryleden voor de inzichtelijke vragen. Ik ben ervan overtuigd dat de tekst dankzij deze opmerkingen consistenter en duidelijker geworden is. De tekst zou niet zo foutloos geweest zijn zonder het grondige naleeswerk van mams, Frederika en Stijn. Bedankt voor jullie tijd!

E.H. M. Ghijs, je bent er niet meer maar ik ben er van overtuigd dat je dit ergens wel zal lezen. Bedankt voor de fijne jaren in *Cantate Domino*. De culturele rijkdom die ik daar meegekregen heb via de muziek en de concertreizen hebben een blijvende impact gehad op niet alleen mijn leven maar ook dat van honderden anderen. Je was een man met een missie en je bent ongetwijfeld de meest begeesterde persoon die ik ooit heb gekend.

Mémé, je staat aan de bron van een hele generatie kinderen, kleinkinderen en ondertussen ook achterkleinkinderen. Van jongsaf heb je ieder met de paplepel '*Hard werken en goed studeren!*' meegegeven. Ondanks je gezegende leeftijd, zit je zelf nog steeds geen vijf minuten stil, want er is altijd nog wel iets te doen om te helpen. Ik ben terecht trots om jou mijn mémé te mogen noemen!

Lieve schat, de laatste maanden zijn de termen *PhD-widow* en *PhD-orphans* een bekend begrip geworden bij ons, bedankt om me in alle rust aan mijn doctoraat te laten werken! Ik heb het nooit moeten vragen, je wás er gewoon steeds waar nodig. Je hebt een rustige vakantie veel meer verdiend dan mij en daar gaan we de komende weken dan ook samen ten volle van genieten!

En mijn twee kleine lieve schatten Joren en Anaïs, bedankt om steeds de dag te beginnen en te eindigen met een zonnetje! Jullie waren er nog niet toen ik begonnen ben met dit doctoraat en kijk nu eens! Joren, wat ben je al een flink kereltje geworden. Wij gaan binnenkort samen eens kamperen, in de bomen klauteren en marshmallows smelten op het kampvuur. Anaïs, mijn kleine meid, ik weet niet hoe het komt, maar als jij lacht, is heel de wereld blij.

en nu . . .

op naar de volgende uitdaging!

Tim.

It is better to know some of the questions
than all of the answers.

- James Thurber.

Abstract

Inferring comprehensive regulatory networks from high-throughput data is one of the foremost challenges of modern computational biology. As high-throughput expression profiling experiments have gained common ground in many laboratories, different techniques have been proposed to infer transcriptional regulatory networks from them and much effort goes to the development of algorithms that infer the structure of transcriptional regulatory networks from this data. In this thesis, the large scale application of simulated gene expression data on network inference algorithms is evaluated and also a novel biclustering model is proposed within the framework of Probabilistic Relational Models.

In the first part of this thesis, a model, called SynTReN, is proposed for generating simulated regulatory networks and associated simulated microarray. This model addresses some of the limitations of previous implementations. Instead of using random graph models, topologies are generated based on previously described transcriptional networks, thereby allowing a better approximation of the statistical properties of real biological networks. The computational performance of our simulation procedure is linear in function of the number of genes, making simulation of large networks possible. The results show the added value of synthetic data in revealing operational characteristics of inference algorithms which are unlikely to be discovered by means of biological micro-array data alone.

The second part of the thesis focuses on the description of an abstracted model of transcriptional regulation, namely by means of a biclustering model. We propose a probabilistic approach to identify overlapping regulatory modules, called *Pro*Bic, based on the framework of Probabilistic Relational Models. The model naturally deals with missing values and noise and thereby leads to a robust identification of biclusters. Both global and query-driven biclustering are combined within a single model-based approach that allows simultaneous identification of multiple and potentially overlapping biclusters. The powerful combination of Probabilistic Relational Models with an Expectation-Maximization approach allows *Pro*Bic to be easily extended to incorporate additional data sources, ultimately leading to the identification of regulatory

modules with associated condition annotation, regulatory motifs and transcription factors.

Korte Inhoud

De identificatie van uitgebreide regulatorische netwerken op basis van hogedoorvoer data is een van de belangrijkste uitdagingen van de moderne computationele biologie. In vele laboratoria worden grote hoeveelheden microrooster gegevens gegenereerd en verschillende technieken zijn op basis hiervan ontwikkeld voor het identificeren van regulatorische netwerken. In dit proefschrift wordt de grootschalige toepassing van gesimuleerde genexpressie gegevens voor het karakteriseren van netwerkinferentie-algoritmen beoordeeld en wordt tevens een nieuw biclustering model voorgesteld binnen het kader van Probabilistische Relationele Modellen.

In het eerste deel van dit proefschrift wordt een simulator beschreven, genaamd SynTReN, voor het genereren van gesimuleerde regulatorische netwerken en de bijhorende gesimuleerde microrooster data. Deze simulator vermijdt enkele van de beperkingen van eerdere implementaties. In plaats van random graaf modellen, worden de netwerktopologieën gegenereerd op basis van eerder beschreven transcriptionele netwerken, waardoor een betere benadering van de statistische eigenschappen van echte biologische netwerken wordt verkregen. Ten tweede, schaalt de computationele kost van onze simulator lineair in functie van het aantal genen, waardoor simulatie van grote netwerken met duizenden genen mogelijk wordt. De resultaten wijzen op de toegevoegde waarde van het gebruik van gesimuleerde gegevens voor het identificeren van operationele kenmerken van inferentie-algoritmen die hoogstwaarschijnlijk niet ontdekt zouden zijn door middel van biologische microrooster data alleen.

Het tweede deel van dit proefschrift richt zich op de beschrijving van een abstract model voor transcriptionele regulatorische netwerken, namelijk door middel van een biclustering model. Dit model, genaamd *Pro*Bic, is ontwikkeld binnen het kader van Probabilistische Relationele Modellen en richt zich op het simultaan identificeren van meerdere overlappende regulatorische modules. Het model behandelt ontbrekende waarden en ruis op een natuurlijke manier en leidt daarmee tot een robuuste identificatie van biclusters. Zowel globale als query-gedreven biclustering worden gecombineerd binnen één enkel modelgebaseerde aanpak die ook de gelijktijdige identificatie van meervoudige en

mogelijke overlappende biclusters mogelijk maakt. De krachtige combinatie van Probabilistische Relationele Modellen met een Expectation-Maximazation algoritme laten ook toe dat *Pro*Bic gemakkelijk kan worden uitgebreid met betrekking tot aanvullende gegevensbronnen, uiteindelijk leidend tot de identificatie van regulatorische modules met bijbehorende conditie-annotatie, regulatorische motieven en transcriptiefactoren.

Notation

Acronyms

AB network	Albert-Barabási network
BN	Bayesian network
CC biclustering	Cheng and Church biclustering model
cDNA	complementary DNA
CGH	comparative genomic hybridization
CPD	conditional probability distribution
DAG	directed acyclic graph
DAPER model	directed acyclic probabilistic entity-relationship model
DNA	deoxyribonucleic acid
DSF network	directed scale-free network
EM	Expectation-Maximization
ER network	Erdös-Rényi network
GBN	ground Bayesian network
GEM	generalized Expectation-Maximization
GEO	Gene Expression Omnibus
GO	gene ontology
HMM	hidden Markov model
ILP	inductive logic programming
IQRN	inter-quartile range normalization
ISA	iterative signature algorithm
JPD	joint probability distribution
MAP solution	maximum a posteriori solution
MCMC	Monte-Carlo Markov Chain
ML	machine learning
MM	Michaelis-Menten
ORF	open reading frame
PCR	polymerase chain reaction
PER model	probabilistic entity-relationship model
PM	perfect match (for probes of a single-channel microarray)
pre-mRNA	precursor mRNA
PRM	probabilistic relational model
PSSM	position specific scoring matrix
QDB	query-driven biclustering

RNA	ribonucleic acid
SF network	scale-free network
SMD	Stanford Microarray Database
SRL	statistical relational learning
SRM	statistical relational model
SQRN	smallest quartile range normalization
SVD	singular value decomposition
SW network	small-world network
TF	transcription factor
TRN	transcriptional regulatory network

Mathematics

$\#X$	the number of elements in a set: if $X = \{X_1, \ldots, X_N\}$, then $\#X = N$
unq(X)	The set of unique values X_i in a vector $X = X_1, \ldots, X_N$
$iset(B)$	assuming that B is a binary vector with elements $B_i \in \{0, 1\}$, $iset(B)$ is the set of vector indices of B for which the vector element is equal to 1

Bayesian networks

X_i	a variable
X	the set of variables $\{X_1, \ldots, X_n\}$
$Pa(X_i)$	the set of parents for a variable X_i
$MB(X_i)$	the Markov blanket of a variable X_i in a Bayesian network.

Probabilistic relational models

C	a class
c	a specific object of class C
A	an attribute
ρ	a reference slot
$\bar{\rho} = (\rho_1, ..., \rho_n)$	a slot chain, which is a chain of reference slots
Σ	a relational schema
σ_r	a relational skeleton for a relational schema Σ
$\mathcal{R}[C]$	the set of reference slots for a class C in a relational schema
$Val(C.A)$	the domain of values for an attribute A of a class C
$\mathcal{A}[C]$	the set of attributes of a class C
$Dom[C.\rho]$	the domain type of reference slot ρ, namely the class C
$Range[C.\rho]$	the range type of the reference slot ρ, which is the class that ρ is referring to
Pa(C.A)	the set of parents in a PRM model for an attribute A of a class C
S	the dependency structure of a PRM model
θ_S	the parameters associated with a dependency structure S

*Pro*Bic model

a	a single array object
A	the set of all array objects
$x.B$	a vector with binary elements $x.B_i$, each element $x.B_i$ indicates the presence (1) or absence (0) of the bicluster i for the entity x (x can be a gene g or an array a)
e	a single expression object
E	the set of all expression objects
g	a single gene object
G	the set of all gene objects
B_e^i	the dot product of the binary vectors $e.gene.B$ and $e.array.B$ where e represents an expression object. By consequence, $iset(B_e^i)$ is the set of bicluster-indices in the intersection of $e.gene.B$ and $e.array.B$, or formally: $iset(B_e^i) = iset(e.gene.B) \bigcap iset(e.array.B)$. $\#iset(B_e^i)$ is the number of elements in this set.

Contents

Robuuste algoritmes voor de inferentie van regulatorische netwerken op basis van expressiemetingen en biologische prior informatie.

Hoofdstuk 1: Inleiding

Met de introductie van microroostertechnologie [95, 140] startte een nieuw tijdperk in de moleculaire biologie: hoge-doorvoer experimenten kunnen nu de expressiewaarden van duizenden genen meten in één enkel experiment. De ontwikkeling van microroosters heeft op zijn beurt geleid tot een groot aantal andere hoge doorvoer databronnen, algemeen bekend als *omics* data zoals transcriptomics, metabolomics, lipidomics en glycomics.

Met de komst van deze hoge-doorvoer technieken en meer computerkracht, is de studie mogelijk geworden van de complexe interacties tussen verschillende biologische entiteiten, zoals genen, eiwitten en metabolieten. Dit domein heet *systeembiologie*. Het gedrag van zo een biologisch systeem kan men niet uitsluitend beschrijven als de som van regels die de individuele componenten beschrijven. Het zijn vooral de *interacties* tussen deze onderdelen die van cruciaal belang zijn om het gedrag van het volledige systeem te kunnen begrijpen. Genen, eiwitten, metabolieten en andere bestanddelen zijn de elementaire componenten in dergelijk ingewikkeld netwerk van interacties. Het cellulaire gedrag wordt bepaald door dit onderliggend regulatorische netwerk. Vanwege deze holistische benadering is systeembiologie een sterk interdisciplinair gebied dat ligt op de intersectie van verschillende andere domeinen zoals biologie, ingenieurswetenschappen en machineleren.

Hoofdstuk 2: Data integratie

In dit hoofdstuk wordt eerst een overzicht gegeven van studies die de reconstructie van regulatorische netwerken doen uitsluitend op basis van mRNA expressiegegevens. Traditionele methoden voor netwerkinferentie van genexpressie gegevens beschouwen ieder gen als een individuele node in het netwerk en hun doel is om alle individuele interacties tussen deze genen te modelleren. Door ieder gen als een afzonderlijke node te beschouwen, creëert men echter een zeer grote zoekruimte van potentiële netwerken. De meeste van deze methoden hebben daarom uitgebreide vereisten wat betreft de grootte van de benodigde dataset en vereisen vaak postprocessing van de resultaten om bijvoorbeeld een ensemble te genereren van alle mogelijke oplossingen. Echter, voor een bioloog ligt het primaire belang niet zozeer in de reconstructie van de interacties tussen alle genen, maar vooral in de reconstructie van de interacties tussen de belangrijkste componenten van de signaaltransductie, namelijk tussen de regulatoren en doelwit-genen. Door deze conceptuele vereenvoudiging wordt de complexiteit van het inferentieprobleem drastisch gereduceerd [159].

Historisch gezien is een eerste categorie van technieken die abstractie van het onderliggende regulatorische netwerk maken, gericht op de identificatie van genen die aanzienlijke over- of onder-expressie vertonen onder de geteste experimentele condities [10]. Een tweede categorie van technieken is gericht op het clusteren van genen die een vergelijkbaar expressieprofiel vertonen onder *alle* geteste condities. In 2001 hebben Cheng en Church voor het eerst de term *biclustering* gebruikt voor het gelijktijdig clusteren van zowel genen als condities in genexpressie data [29]. Sindsdien zijn verschillende biclustering algoritmes ontwikkeld (zie o.a. [110]) met elk hun eigen focus voor de identificatie van specifieke types biclusters.

Daarnaast is ook een groeiende interesse gekomen in de modulaire beschrijving van regulatorische netwerken [79]. Genen die coexpressed zijn voor een subset van de condities en die gelijkaardige interacties vertonen binnen het regulatorische netwerk, kunnen worden gegroepeerd in een *regulatorische module* [79]. Naast een gelijkaardig expressieprofiel hebben deze genen ook een aantal andere eigenschappen gemeen, zoals een gemeenschappelijke set van regulatoren of een gemeenschappelijk gen-ontologie annotatie.

Door middel van een een modulaire representatie, kunnen alle genen binnen eenzelfde module beschreven worden met dezelfde set van parameters in plaats van met een afzonderlijke set van parameters per gen. Deze reductie van het aantal parameters is niet alleen interessant voor het terugdringen van de complexiteit van het model, maar het biedt ook nieuwe inzichten in de structuur en organisatie van de regulatorische interacties tussen de genen.

Met de beschikbaarheid van heterogene omics gegevens, wordt de complexiteit van het probleem van netwerk- of module-inferentie mogelijk sterk gere-

duceerd. Verschillende omics data ontsluieren verschillende en vaak complementaire aspecten van regulatorische netwerken en de integratie van al deze data levert een vollediger inzicht op in het onderliggende netwerk. Hier zullen we ons richten op hoe goed de verschillende computationele methoden voor inferentie van transcriptionele netwerken kunnen omgaan met de specifieke biologische kenmerken van hoge-doorvoer gegevens. Opgemerkt moet worden dat de methoden beschreven in dit hoofdstuk niet organisme-specifiek zijn hoewel de meeste van hen getest op *Saccharomyces cerevisiae*, het meest uitgebreid bestudeerd modelorganisme [28].

Hoofdstuk 3: Een synthetisch model van transcriptionele regulatie: SynTReN

De inferentie van complexe regulatorische netwerken op basis van hoge-doorvoer data is één van de belangrijkste uitdagingen binnen computationele biologie. Verschillende technieken zijn reeds voorgesteld voor het identificeren van transcriptioneel regulatorische netwerken op basis van deze data. In dit hoofdstuk wordt een model, genaamd SynTReN, voorgesteld voor het genereren van gesimuleerde regulatorische netwerken en bijbehorende gesimuleerde microrooster gegevens. Dit model vermijdt enkele van de beperkingen van eerdere simulatoren o.a. met betrekking tot de maximale grootte van de gesimuleerde netwerken en het opstellen van gesimuleerde experimenten op grote schaal. In plaats random graaf-modellen te gebruiken, worden netwerktopologieën in SynTReN gegenereerd op basis van eerder beschreven transcriptionele netwerken waardoor een betere benadering van de statistische eigenschappen van echte biologische netwerken wordt bekomen. Daarnaast schaalt de rekenkundige kost van de simulatie lineair in functie van het aantal genen waardoor simulatie van grote netwerken mogelijk wordt.

De operationele kenmerken van drie bekende netwerkinferentie-algoritmen worden bepaald, namelijk van ARACNE, Genomica en SAMBA, die elk een verschillende gedrag vertonen in functie van de verschillende parameters van de gesimuleerde gegevens. De geteste parameters waren netwerkgrootte, netwerktopologie, het type en de hoeveelheid ruis, de hoeveelheid beschikbare data en de interactietypes tussen soorten genen.

Experimenten hebben aangetoond dat de onderliggende netwerktopologie een sterke invloed heeft op de prestaties van inferentie-algoritmen, een conclusie waarbij rekening moet worden gehouden bij de evaluatie van inferentie-algoritmes aan de hand van gesimuleerde datasets. Voor twee van de geteste algoritmen, Genomica en ARACNE, zijn de inferentieresultaten beter voor (sub)netwerken op basis van biologische netwerken. Dit geeft aan dat er nog ontbrekende karakteristieken zijn van biologische netwerken die niet door random graaf-modellen worden gemodelleerd.

De bekomen resultaten wijzen op de toegevoegde waarde die gesimuleerde data kan bieden in het bepalen van de operationele kenmerken van inferentie-algoritmen aangezien deze kenmerken hoogstwaarschijnlijk niet geïdentificeerd kunnen worden door middel van biologische microrooster data alleen. Deze resultaten ondersteunen in het algemeen het gebruik van computermodellen binnen het onderzoek in systeembiologie.

Hoofdstuk 4: Probabilistische Relationele Modellen

Met de verhoogde opslag- en verwerkingscapaciteit van de huidige computers en de opkomst van grote online databases met relationele informatie, heeft zich een explosie van beschikbare gegevens voorgedaan. Veel van deze datasets worden opgeslagen in complexe relationele databases, maar de meest gekende algoritmes voor machineleren zoals bvb. Bayesiaanse netwerken [126], k-means clusteren [106], beslissingsbomen [132] of neurale netwerken [22] kunnen niet rechtstreeks worden toegepast op deze relationele datasets omdat ze geleerd worden op basis van gegevens uit een enkelvoudige tabel, ook genaamd *attribuut-waarde* gegevens.

Meer expressieve technieken voor machineleren die zowel variabelen als de relaties tussen deze variabelen kunnen leren, heten *relationele data mining* methoden. Een hernieuwde interesse in relationele data mining heeft in de afgelopen jaren geleid tot een nieuw onderzoeksdomein rond *statistische relationele modellen*. Dit domein ligt op de doorsnede van machineleren, kennisrepresentatie en probabilistische modellen. In dit hoofdstuk richten we ons vooral op een bepaalde klasse van zogenaamde statistisch relationele modellen, namelijk op probabilistische relationele modellen [55, 61, 94]. PRM's zijn toegepast op een verscheidenheid van relationele machine learning problemen [34, 62, 122] en verschillende toepassingen werden ontwikkeld door E. Segal op het gebied van bioinformatica [144, 145, 146, 147]. PRM's bieden een elegante manier voor het beschrijven van een biclustering model dat is makkelijk uitbreidbaar naar de integratie van aanvullende gegevensbronnen zoals nader besproken wordt in Hoofdstuk 5.

In dit hoofdstuk wordt verder een korte introductie gegeven met betrekking tot Bayesiaanse netwerken en over hoe deze netwerken kan leren in geval van complete en incomplete data. Twee vaak gebruikte voorwaardelijke kansverdelingen (VKV), namelijk tabel VKV's en Gaussiaanse VKV's, werden gedefinieerd waarmee zowel discrete als continue data kunnen worden gemodelleerd. In het belangrijkste deel van dit hoofdstuk wordt de definitie van PRM's en hun relatie tot Bayesiaanse netwerken uitgelegd. Een fictief voorbeeld over *Influenza* infecties en een set patiënten die verschillende behandelingen krijgen, geven aan hoe PRM's gebruikt kunnen worden om concepten binnen dit relationele domein in een probabilistisch model te gieten. Het

leren van PRM's in geval van complete en incomplete data wordt gerelateerd met de eerder geïntroduceerde concepten voor Bayesiaanse netwerken. Het *Expectation-Maximization* algoritme werd specifiek belicht als een interessante leertechniek voor PRM's in geval van incomplete data.

Hoofdstuk 5: *Pro*Bic model

Het tweede grote luik van dit proefschrift richt zich op de beschrijving van een geabstraheerd model van transcriptionele netwerken, namelijk door middel van een biclustering model. Een probabilistisch model, genaamd ProBic, werd voorgesteld voor het simultaan identificeren van overlappende regulatorische modules binnen het framework van Probabilistische Relationele Modellen.

De identificatie van transcriptioneel regulatorische netwerken op basis van genexpressie gegevens is een zeer actief gebied van onderzoek. Het is echter ook een ondergedetermineerd probleem omdat het aantal mogelijke interacties en hun geassocieerde parameters veel groter zijn dan de dimensionaliteit van de beschikbare gegevens. Bovendien bevatten de huidige microrooster gegevens inherent veel ruis. Veel technieken zijn daarom ontwikkeld om robuuste representaties van het onderliggende netwerk te genereren door een reductie van het aantal parameters, vaak gerealiseerd door groepering van genen en/of condities in regulatorische modules.

Door het gebruik van een probabilistisch kader voor *Pro*Bic, worden ontbrekende waarden en ruis op een natuurlijke manier gemodelleerd, wat leidt tot een robuuste identificatie van biclusters onder verschillende instellingen van ruis en de ontbrekende waarden. Zowel globale als query-gedreven biclustering worden gecombineerd binnen één enkel model-gebaseerde biclustering methode. Een reeks van experimenten op een compendium van *Escherichia coli* microrooster gegevens [102] hebben aangetoond dat de query-gedreven biclustering in staat is gebruik te maken van queries met enkelvoudige genen, een eigenschap die niet wordt gedeeld door alle query-gedreven biclustering algoritmes. Een tweede reeks experimenten op het *E. coli* compendium hebben bovendien aangetoond dat *Pro*Bic robuust is met betrekking tot outlier genen binnen een set van query-genen.

Tot slot laat de krachtige combinatie van Probabilistische Relationele Modellen met een Expectation-Maximization strategie toe dat *Pro*Bic gemakkelijk kan worden uitgebreid met additionele gegevensbronnen, uiteindelijk leidend tot de identificatie van regulatorische modules met bijbehorende conditie-annotatie, transcriptiefactoren en de geassocieerde regulatorische motieven.

Hoofdstuk 6: Conclusie en toekomstperspectieven

Dit hoofdstuk vat de belangrijkste onderzoeksresultaten samen en stelt ook een aantal uitbreidingen voor wat betreft toekomstig onderzoek binnen dit domein.

Deel I:

- Een netwerk-generator en simulator werd ontworpen die in staat is grote regulatorische netwerken met duizenden genen te simuleren. De huidige state-of-the-art dynamische simulatoren simuleren netwerken slechts tot maximaal een paar honderd genen. Door uitsluitend steady-state oplossingen te beschouwen, kan de simulatie van een netwerk met duizenden genen computationeel berekenbaar gemaakt worden.

- Terwijl inferentie-algoritmen vaak worden getest op gesimuleerde data, wordt de topologie van het onderliggende netwerk vaak niet als belangrijke factor in rekening gebracht. Onze resultaten tonen echter aan dat de keuze van netwerk topologie voor de gesimuleerde data een grote impact heeft op de kwaliteit van de inferentie voor de geteste inferentie-algoritmen.

- Verschillende inferentie-algoritmen werden toegepast op gesimuleerde datasets met elk verschillende kenmerken. De resultaten tonen een kwalitatief zeer verschillende respons van de algoritmen met betrekking tot de parameters van de gesimuleerde data zoals hoeveelheid ruis, de hoeveelheid gegevens en de types interacties tussen de genen. Deze resultaten tonen aan dat gesimuleerde data inzicht in de operationele kenmerken van een algoritme oplevert die complementair zijn aan de inzichten op basis van biologische gegevens alleen.

Deel II:

- Een efficiënt biclusteringsalgoritme, genaamd *Pro*Bic, is ontwikkeld in het kader van probabilistische relationele modellen, dat geen voorafgaande discretizatie vereist van de expressiemetingen.

- Het biclusteringsmodel behandelt door zijn probabilistische aard ontbrekende waarden en ruis op een natuurlijke manier, leidend tot een robuuste identificatie van biclusters onder verschillende instellingen van ruis en de ontbrekende waarden.

- Zowel globaal als query-gedreven biclusteren kunnen gecombineerd worden binnen één enkele modelgebaseerde aanpak. De query-gedreven aanpak is ook robuust gebleken met betrekking tot zogenaamde 'outliers' in de set van query genen.

- *Pro*Bic identificeert tegelijkertijd meerdere overlappende biclusters en een uitbreiding van *Pro*Bic laat ook toe om zowel gecorreleerde als anti-gecorreleerde genen te groeperen binnen één enkele bicluster.
- De krachtige combinatie van PRM's met een Expectation-Maximization algoritme, laten toe om *Pro*Bic op een eenvoudige manier uit te breiden om additionele databronnen te incorporeren, ultiem leidend tot de identificatie van regulatorische modules met een geassocieerde conditie annotatie, regulatorische motieven en transcriptiefactoren.

Naar de toekomst toe zien we twee belangrijke uitdagingen. Enerzijds is er met de beschikbaarheid van steeds meer heterogene omics data en de ontwikkeling van de integratieve algoritmen die dergelijke datasets combineren, een behoefte aan meer volledige gesimuleerde modellen die naast transcriptionele regulatie ook alle andere interacties tussen DNA, RNA, eiwitten en metabolieten modelleren. Anderzijds is *Pro*Bic ontworpen om te worden uitgebreid naar de identificatie van cis-regulatory modules. De voorgestelde uitbreidingen uit Hoofdstuk 5 zijn dan ook een interessant startpunt voor verder onderzoek.

Chapter 1

Introduction

1.1 History

Figure 1.1: ***Gregor Mendel.***

In 1866, only seven years after the publication of *On The Origin of Species* by C. Darwin [35], the foundations of genetics and the rules that govern its transmission were laid by G. Mendel [15] (Figure 1.1) in an almost completely ignored publication about breeding experiments on *Pisum sativum* (garden pea). From the results of these experiments, Mendel derived two basic rules that governed the inheritance of different traits of the garden pea. It was only well after his death that the importance of his work was recognized. Three

years after the publication of Mendel's experiments, F. Miescher discovered deoxyribonucleic acid (DNA) [117] . Its biological function however, remained unknown for decades.

Despite these major developments in the 19^{th} century, no significant breakthroughs were made in the domain of genetics for almost a century. It was only in 1944 that Avery, MacLeod and McCarty identified DNA as the substance responsible for genetic transformations in a milestone experiment [7]. In 1953, the work of Franklin [8] led to the discovery of the double helix structure of DNA by J. Watson and F. Crick [165]. This groundbreaking discovery suggested how DNA replication occurred, how hereditary traits are inherited and how they would undergo mutations. In the years that followed, further experiments unraveled the exact mechanisms by which these processes occurred. These two discoveries sparked the start of *molecular biology* as a new field in science. It also led to the formulation of what is known as the *central dogma*[1] (1958) of molecular biology: the information flow in an organism is carried out from DNA to ribonucleic acid (RNA) to protein (see also Section 1.2).

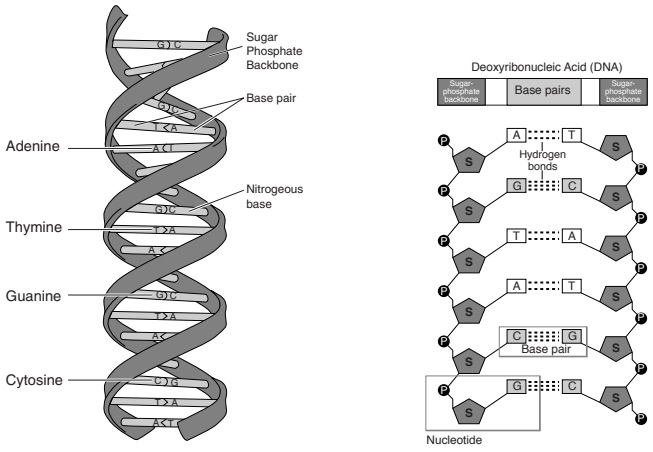
With the introduction of microarray technology [95, 140] in the '90s, a new era started in molecular biology. For the first time in history, high throughput experiments measured the expression levels of thousands of genes simultaneously in a single experiment. Microarrays quickly gained interest of a large community of scientists ranging from biologists to physicians. The explosion of data in molecular biology led to the introduction of a new domain that uses advanced computational methods to deal with these datasets, namely *computational biology*.

Microarrays were the first members of a large number of high throughput techniques that generated data in a wide variety of domains. These domains and the type of data that are linked to them, are now commonly known as *omics* and include transcriptomics, metabolomics, lipidomics, glycomics, spliceomics, pharmacogenomics and many others. Vast amounts of heterogenous data have since been gathered in public databases all over the world (for example: Entrez, GenBank, UniProt, Ensembl, TRANSFAC, KEGG, ArrayExpress, GEO, ...) and the first efforts for combining these data emerged [27, 68, 79, 80]. The introduction of these high throughput *omics* data again led to a new research field, called *systems biology*, that studies the interactions between different entities in biological systems and how these *interactions* lead to the functioning of the complete system rather than studying the individual components in isolation from their environment (see Section 1.4).

[1] This *central dogma* proved to be an oversimplification of biology. It was restated accordingly by Crick in 1970 and included information transfer from DNA to DNA, RNA to RNA, RNA to DNA and DNA to protein [33]. The discovery of other regulation mechanisms like RNA interference, of epigenetic phenomena such as DNA methylation and many other mechanisms have added even more complexity and exceptions to these rules.

1.2 Molecular biology

In this section, we introduce the basic concepts of molecular biology. One of the key molecules for all living organisms is *DNA*, as it is the carrier of its genetic information. The DNA molecule is composed out of two complementary strands and forms a double helix structure (see Figure 1.2) and each of the strands is a chain of nucleotides. Four types of nucleotides exist in all living organisms on this planet and each nucleotide has the same chemical structure: it consists of a sugar, a phosphate group, deoxyribose and one of the following four bases: *adenine* (A), *cytosine* (C), *guanine* (G) and *thymine* (T). Because of their molecular structure, these bases only bind in pairs by means of a hydrogen bond: cytosine only binds to guanine and adenine only binds to thymine, a process called *complementary base pairing* [67]. This base pairing holds the two complementary DNA strands together. Each of the complementary DNA strands has a specific direction due to the molecular asymmetry in the nucleotides. The two ends of a DNA strand are labeled with 5′ and 3′ labels. Note that, by convention, the DNA code is represented from 5′ to 3′.



(a) DNA structure (b) Nucleotide base pairing

Figure 1.2: ***Illustration of DNA structure and nucleotide base pairing.*** *(a) DNA double helix structure. (b) Detail of the DNA double helix structure and complementary base pairing. The four nucleotides only bind in pairs: guanine (G) pairs with cytosine (C) and thymine (T) pairs with adenine (A).* [figures from http://www.genome.gov/glossary.cfm]

1.2.1 Central dogma in molecular biology

In biological systems, proteins are the workhorses that perform a wide range of functions such as catalysation of biochemical reactions, gene regulation,

cell signaling and immune responses, as well as providing structural and transportation functions. The central dogma (see Figure 1.3) describes how proteins are formed, starting from DNA.

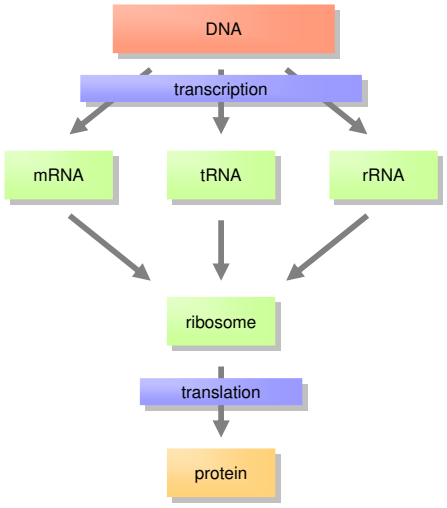


Figure 1.3: ***Illustration of the central dogma in molecular biology****. The central dogma states that information in biological systems is passed from DNA to RNA (*transcription*) and from RNA to proteins (*translation*).*

The first step in protein synthesis is the *transcription* of a specific part of the DNA that lies between a start and a stop codon to a messenger RNA (mRNA). A *codon* is a set of three nucleotides that either encode for a specific amino acid or indicate the beginning (start codon) and ending (stop codon) of an open reading frame (ORF). The two DNA strands are separated at the ORF starting point and one of the strands is used as a template from which the precursor mRNA (pre-mRNA) is transcribed. This pre-mRNA is optionally[2] *spliced*, during which introns are removed from the pre-mRNA and exons are joined to form the mature mRNA as illustrated in Figure 1.4.

In the second step, called *translation*, the resulting mRNA is translated by means of the ribosomes into a peptide chain consisting of a series of *amino acids*. The mRNA is scanned per codon. There is a large redundancy in the genetic code, since each possible combination of three nucleotides leads to 64 combinations, but only 20 possible amino acids are encoded with these combinations (actually only 62 combinations are available as the start and stop codon require two encodings). One of the beneficial consequences of the redundancy in the genetic code is the increased fault-tolerance for point

[2]Introns do not exist in prokaryotic genomes, so splicing only occurs in eukaryotes.

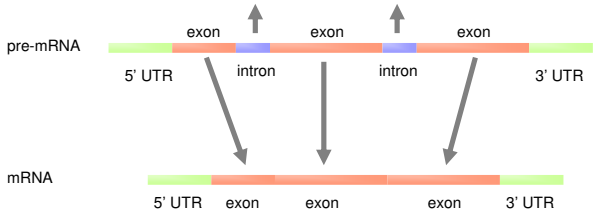


Figure 1.4: ***Splicing of precursor mRNA into mature mRNA****. The pre-mRNA consists of a number of introns and exons between the 5′ and 3′ UTRs. The introns are spliced out of the pre-mRNA and the remaining exons are joined and form the resulting mRNA together with the 5′ and 3′ UTRs.*

mutations. A graphical illustration of the transcription and translation steps is given in Figure 1.5.

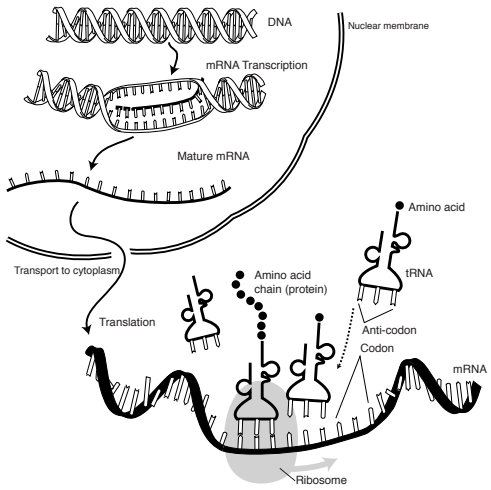


Figure 1.5: ***Transcription and translation in eukaryote cells****. In the nucleus, DNA is transcribed into mRNA (optionally by means of an intermediate step where pre-mRNA is spliced to form mRNA). This mRNA is transported outside the nucleus into the cytoplasm where it is processed by the ribosomes (translation step). The ribosomes translate each set of three nucleotides into an amino acid by means of tRNA (transfer RNA) and attach it to a growing peptide chain which will subsequently fold into a protein.* [figure from http://www.genome.gov/glossary.cfm]

During and after the translation step, the peptide chain folds into the resulting protein during the *protein folding* step. The protein can be changed further, e.g. through *post-translational modifications* like phosphorylation, to obtain its final

physicochemical structure.

Some proteins, called *transcription factors* (TFs), regulate the rate at which genes produce mRNA by binding to specific target sites in the genome. These target sites are short sequences of nucleotides and are mainly located in the promotor region or the *cis*-regulatory region of the gene, which is illustrated graphically in Figure 1.6. Target sites of genes that are regulated by a common TF, often have common characteristics and can be represented by a regulatory *motif*. This is a probabilistic description of the common nucleotide structure of different target sites. The complete set of transcriptional interactions between all genes and their transcription factors can be visualized as a network and is called a *transcriptional regulatory network* [9, 101, 109, 149].

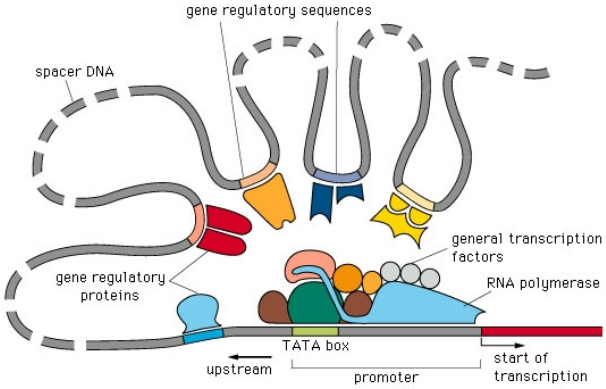


Figure 1.6: ***Illustration of transcriptional gene regulation mechanisms.***

Transcriptional regulation is considered the predominant factor for the control of gene expression [72] and is therefore an important component of the complete cellular signaling and regulation network. Other mechanisms that directly or indirectly affect gene regulation have also been identified and include for example RNA interference [100], post-translational modifications, epigenetic factors such as DNA methylation [21, 104], protein-protein interactions and metabolic interactions.

1.3 High-throughput techniques

With the introduction of microarrays [95, 140], a new era started in molecular biology: high throughput experiments could now measure the expression levels of thousands of genes in a single experiment. By performing experiments under different conditions or at different time points, biologists can now monitor the transcriptional behavior of all these genes simultaneously. The introduction of microarrays has led to the development of a large number of

other high throughput data types, commonly known as *omics* data (e.g. transcriptomics, metabolomics, lipidomics, glycomics, etc.). We will only discuss DNA microarrays (single channel and dual channel) here, as this forms the historical basis of all other techniques.

Microarray technology is widely used in many different research areas such as comparative genomic hybridization (CGH) [129], gene expression analysis [140], transcription factor binding [136, 170] and DNA methylation [141].

A *DNA microarray* (or DNA chip) is a collection of microscopic spots on a solid surface that are organized in a matrix structure. On each spot, single stranded DNA strains with a specific sequence are attached and selectively bind to their complementary DNA strains. These DNA strains are called *probes* and when the microarray is exposed to a biological sample during an experiment, these probes will therefore selectively bind particular complementary DNA (cDNA) strands from the sample. The cDNA material in the sample is labeled with one or more fluorescent dyes and after the experiment, one or more images of the microarray are taken under laser light of different wavelengths. An example of the resulting image is shown in Figure 1.7.

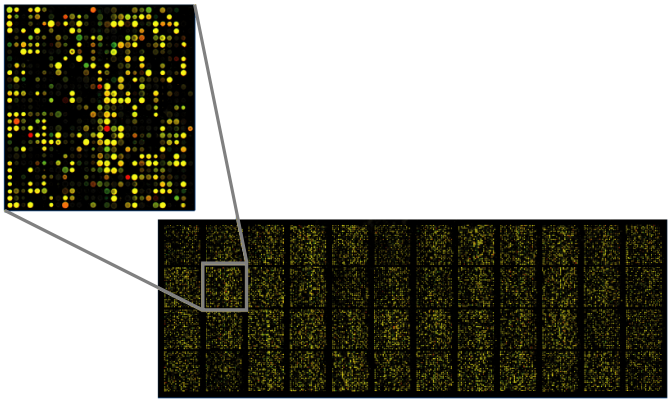


Figure 1.7: ***Example of a two-channel microarray image.***

Two main technologies exist for fabrication of DNA microarrays, namely *cDNA microarrays* and *oligonucleotide microarrays*. In *cDNA microarrays*, probes are 'spotted' onto a glass substrate using an array of fine needles. The probes are either long (100-1000 bases) complementary DNA strains that bind to particular DNA sequences of interest or presynthesized long oligonucleotide probes, which are typically 50-80 bases. In *oligonucleotide microarrays*, the probes are short oligonucleotides (10-30 bases), which are typically synthesized using polymerization techniques. DNA microarrays can also be divided into two categories based on the number of channels or dyes that are used: single-channel and two-channel microarrays. Single-channel arrays use one single

dye for labeling biological samples and each sample is hybridized onto a different microarray. Two-channel arrays use two different dyes (one per sample) and these differently labeled samples are then hybridized onto the same microarray. Both categories are outlined in the sections below.

1.3.1 Two-channel microarrays

Two-channel microarrays use two dyes to color the genetic material. In two-channel arrays, both cDNA or long oligonucleotides can be used for the probe design. cDNA microarrays were often used in academia due to their lower cost. However, the technique has some major disadvantages and is often replaced now by long oligonucleotide arrays. The typical length of these long oligonucleotides is 60 nucleotides, leading both to a sufficient degree of hybridization and a high sensitivity and specificity [75].

We will outline the procedure for gene expression profiling using two-channel microarrays here using long oligonucleotide microarrays. The procedure, outlined in Figure 1.8, is very similar to the cDNA microarray procedure since both techniques are based on two-channel arrays. For oligonucleotide arrays, a specific set of probes is designed in which each probe targets a specific gene of interest and for cDNA arrays, these probes are derived from biological samples.

In the first step, two biological samples are prepared. Often these samples are a reference sample and a sample of interest (for example normal tissue versus cancer tissue). However more complex experiment designs can be set up such as loop designs or saturated designs, depending on the type of experiment and the desired analysis. After the preparation of the samples, mRNA is extracted, purified and amplified from both these samples. Optionally, the less stable mRNA can be reverse transcribed into more stable cDNA.

In the next step the cDNA(RNA) is labeled with a different fluorescent dye for each of the biological samples. Typically the dyes *Cy3* and *Cy5* are used, which light up green and red respectively under specific wavelengths of light. The two differently labeled samples are then joined in a single solution and the microarray is covered with this solution. The cDNA(RNA) hybridizes selectively with the probes on the microarray that have a complementary sequence as the cDNA(RNA). After washing away the non-hybridized material, laser light of specific wavelengths is used to illuminate the fluorescent dyes and an image is taken of the microarray.

This image contains spots of essentially four colors: black, green, red and yellow. A spot will be black, green, red or yellow in this image if binding occurred respectively with either none of the samples, only sample 1, only sample 2 or both samples. For each spot, the spot's shape and size together with the intensity distribution for both dyes in the spot are then measured together with the background intensity of the image. These raw data are subsequently

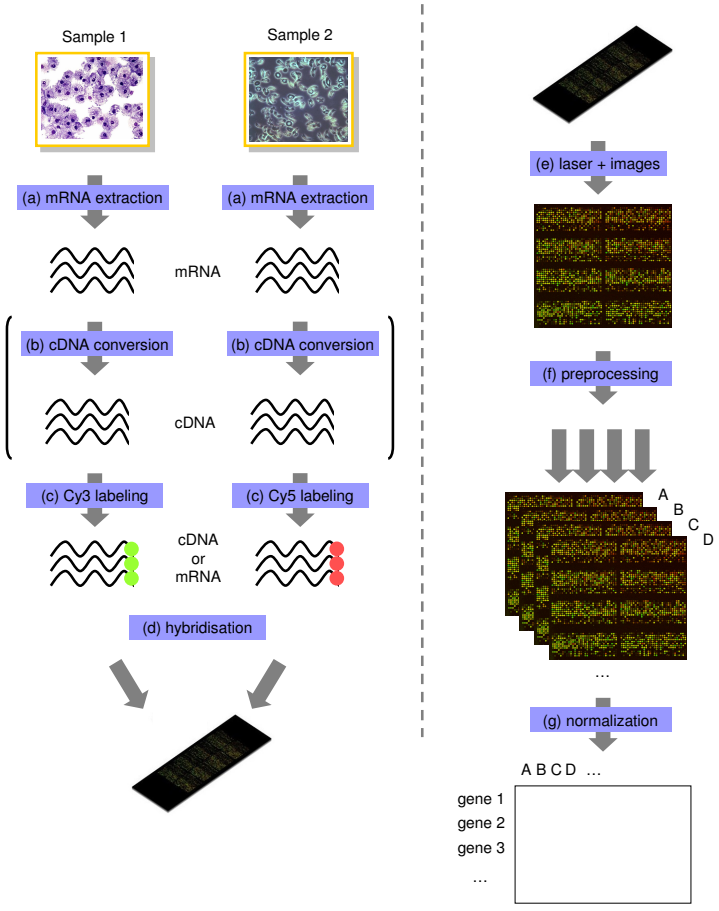


Figure 1.8: ***Gene expression profiling process overview using two-channel microarrays.*** ***(a)*** *mRNA is extracted from two samples.* ***(b)*** *Optionally the mRNA is reverse transcribed into cDNA, which is more stable than mRNA.* ***(c)*** *The cDNA(RNA) in each of the samples is labeled with a different fluorescent dye, typically* Cy3 *and* Cy5 *dyes are used which light up green and red respectively under specific wavelengths of (laser) light.* ***(d)*** *Both cDNA(RNA) extracts of the samples are hybridized with the probes on the microarray.* ***(e)*** *Laser light of specific wavelengths is shone on the microarray, illuminating the fluorescent dyes. In the resulting image, a spot will be black, green, red or yellow if binding occurred respectively with none of the samples, only sample 1, only sample 2 or both samples.*

processed in sometimes complex preprocessing and normalization steps in order to reduce noise and artifacts from each of the steps in the experiment.

Preprocessing involves the elimination of systematic noise sources such as

array effects, plate effects and pin effects for cDNA microarrays so that the remaining variation in the data is maximally correlated with the underlying biological effects. The main steps involve a quality assessment step which is often performed visually and a background correction step. A normalization procedure is then applied to calibrate the microarray data by correcting for dye effects and probe effects. This results in a set of probe intensity values are for both dyes. For gene expression profiling, probes need to be linked with genes. The cDNA microarrays are often designed to have an almost one-to-one mapping between probes and genes.

1.3.2 Single-channel microarrays

The second class of DNA microarrays is the short oligonucleotide array. This type of microarray is always single-channel (i.e. it uses one dye). The platform of Affymetrix (http://www.affymetrix.com) is the most widely used short oligonucleotide platform and we will discuss single-channel microarrays by means of this platform. An illustration of an Affymetrix single-channel microarray is given in Figure 1.9.

Figure 1.9: ***Example of an Affymetrix GeneChip****. The GeneChip Human Genome U133 contains more than 54,000 probe sets and 1,300,000 distinct spots, covering the expression level of virtually all human genes.*

Short oligonucleotides are synthesized on a substrate (also called *chip* or *slide*) using a similar technique as for the production of integrated circuits, namely through photolithography. The process is outlined in Figure 1.10. The chip is initially covered with *linker molecules*. When such a linker molecule is exposed to light, it is activated and it will bind a nucleotide. An iterative procedure is now applied in which part of the chip is covered with a photoresistant mask. When light is shone on the unprotected sites, the linker molecules are activated. The chip is then covered with a solution containing a single oligonucleotide

which will bind to the unprotected sites. The solution is washed away and the mask is removed, after which the whole procedure is repeated for a different mask and nucleotide solution. The result of this procedure is a chip with on each site a probe of a specific length (typically 20-25 bases).

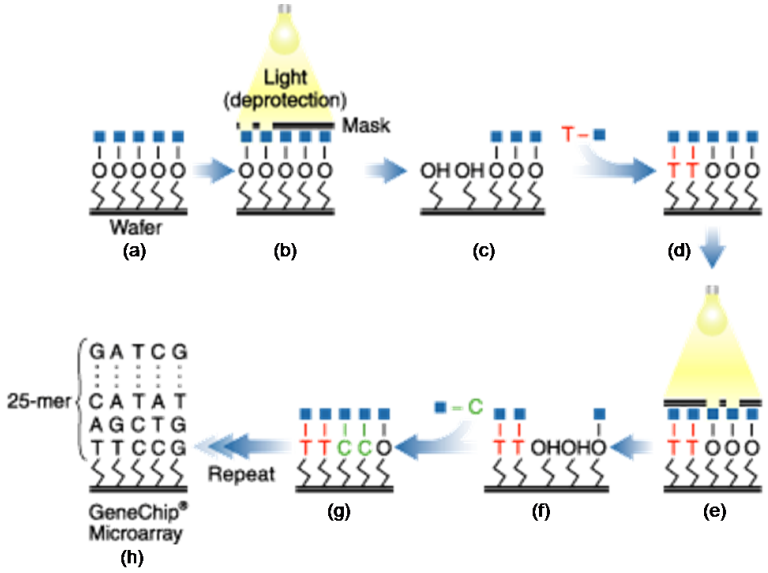


Figure 1.10: ***Photolithographic procedure for manufacturing of Affymetrix GeneChip.*** ***(a)*** *the chip surface is coated with 'linker' molecules.* ***(b)*** *a photoresistant mask is applied to shield part of the linker molecules and ultraviolet light is shone over the mask.* ***(c)*** *the unshielded linker molecules are activated.* ***(d)*** *a solution containing a single oligonucleotide covers the surface of the chip and the nucleotide attaches to the activated areas on the chip.* ***(e)-(g)*** *the procedure (2)-(4) is repeated with different nucleotide solutions until the probes reach their desired length (usually 20-25 bases).* ***(h)*** *this leads to a completed GeneChip with unique probes on each spot.* [image from Affymetrix http://www.affymetrix.com/technology/manufacturing/index.affx]

Due to their shorter length, these probes are not as specific as for the two-channel arrays. A gene is now not represented by a single probe, but rather by a *probe set*, typically consisting of 10-20 *probe pairs*. For each probe pair, one probe matches with a part of the sequence of the gene, called a *perfect match* and another probe has the same sequence except for one nucleotide in the middle of the probe. The latter probe is called the *mismatch*. The difference in binding between these two probes is a measure for the degree of binding of the gene of interest. For a single-channel array, each of the biological samples are now colored with the same dye and each of the samples is hybridized on a different array.

Single-channel microarrays require completely different preprocessing and normalization steps. The two most widely used techniques are Microarray

Suite 5.0 (MAS 5.0) and Robust Multi-array Average (RMA). For an overview of these techniques, we refer to the Microarray Suite User Guide [1] and [82] respectively. The two main differences between single- and two-channel arrays are the absence of a dye effect in single-channel arrays, avoiding a bias in the results due to the different dye intensity and saturation effects. Secondly, the smaller probe length and the specific probe design in single-channel arrays, requires specific normalization techniques.

1.4 Self-organization and systems biology

Dynamic systems in physics and in other domains can exhibit complex, chaotic and self-organizing behavior. Such complex behavior can emerge from simple interacting physical elements. An analogy that is often used to describe these systems is by means of an ant colony whose complexity emerges from the interaction between its 'simple' constituents, the ants. The isolated behavior of an ant can in principle be described through a set of simple rules like *'if near food, then lift food'* or *'if smell pheromone then walk to pheromone source'*. However, the *interaction* between many of these individual ants results in a highly complex system, namely the ant colony, that *emerges* from the simple isolated behavior of the ants. This emergent behavior can often not be predicted from the isolated behavior of the single ants in the system, a property called *strong emergence* [96]. Examples of such higher levels of organization are for example the complex maze of tunnels with air-conditioning pipes that is built by the worker ants and the organized food foraging and storage system.

Analogous to an ant colony, a biological cell also exhibits emergent behavior. The individual entities are genes, proteins, metabolites and many other entities whose interactions are described with relatively simple rules, e.g. *'gene A is active if transcription factor X is bound'*. The behavior of the complete system is not merely the sum of the rules that govern its constituents. The *interactions* between these constituents are in fact crucial to understand the behavior of the system. The analogy with an ant colony also illustrates that studying the isolated behavior of each individual component does not always reveal the higher level organization of the system.

With the advent of high throughput techniques and more computational power, the study of complex interactions between biological entities, such as genes, proteins and metabolites, has become possible and is called *systems biology*. Rather than using a traditional reductionist approach where individual entities and small biological pathways are studies in isolation, systems biology adopts a holistic approach that studies the biological system as a whole. Genes, proteins, metabolites and other constituents are the basic entities in this complicated network of interactions. The cellular behavior is mediated by this underlying regulatory network. Because of this holistic approach, systems

biology is a highly interdisciplinary field that lies at the intersection of several other domains like engineering, biology and data mining.

Since systems biology aims at obtaining a global insight into the biological system, the integration of different high-throughput data sources is important in the top-down systems biology approach. Heterogeneous data sources describe the same biological system from a different point of view and combining these data can as such help obtaining a holistic view on the system.

Focussing on the engineering aspects of systems biology, researchers are now approaching biological systems directly from an engineering perspective: artificial gene regulatory circuits have been built with specific functions like *toggles* [57] or *oscillators* [50], much like electronic circuits are being designed by engineers. Recently, even more ambitious links between engineering and biology are explored: in *synthetic biology* [18] complex biological systems are being built using standard, interchangeable parts, called *devices*. These devices consist of one or more biochemical reactions like transcription or translation. A growing number of these devices are stored in a central repository at MIT (http://parts.mit.edu). These basic parts are then combined to perform a complex engineered behavior.

While the analogies between engineering and synthetic biology are more clear, there are some fundamental differences in how systems biology approaches the investigation of a complex system. Engineers use a highly structured approach to construct or investigate for example a radio. A power unit is connected to an electronic circuit and an antenna is used to receive incoming signals. Some buttons and a display provide the interface with the outside world by which the radio is controlled. Each of these high-level components further consists of a number of subcomponents with a specific function and this process continues until we reach some components with basic functionality like a transistor. The engineer has a complete knowledge and insight in the system's behavior at every abstraction level.

In contrast to the engineer's radio, the 'radio' system is not constructed but a given in systems biology, namely a biological cell. To understand its inner mechanics, the biological system needs to be examined by means of a series of experiments. In analogy to a radio system, such experiments would be similar to breaking the radio in pieces and cataloguing the pieces or selectively damaging a single component and checking the resulting effect on the system. A nice analogy is described in [97] between the engineer's and the biologist's approach for fixing a broken radio.

The number of required experiments to gain complete insight in the underlying system is extremely high. To illustrate the complexity of the problem: a single human cell contains about 20,000-25,000 genes [81]. About 500 million interactions could potentially exist between each pairs of genes. If a systems biologist desires to infer both the presence and the strength of each interaction, a huge amount of data is required for each of these interactions. Even with our

current basic understanding of the inner workings of a cell, it is computationally infeasible to derive the complete interaction network from the available data.

Moreover, only 1.5% of our entire genome is transcribed into proteins and the function of the other 98.5% DNA is considered unknown at this moment. This DNA used to be called 'junk DNA', however a substantial fraction of this non-genic DNA is in fact transcribed into (non-coding) RNA [31]. There are clear indications that at least some of this non-coding RNA also has a regulatory function, thus adding even more to the complexity of the regulatory system.

1.5 From biology to modeling

A living organism is a tremendously complex piece of machinery. Each cell in an organism exchanges information with its neighboring cells and adapts its behavior accordingly. Within each cell, there are different entities such as genes, proteins, metabolites and RNA that interact with one another, form complexes and catalyze reactions. Figure 1.11 gives a schematic overview of the interactions within a biological cell and a number on different views on this set of interactions. Each of the views corresponds to a specific biological entity for which data can be collected using a number of high-throughput techniques and it is associated with an *omics* domain such as genomics, transcriptomics, proteomics or metabolomics.

The mere collection of these large datasets does not provide us with insight in the underlying mechanisms that generated the data. To extract useful knowledge from these datasets, *models* need to be built that provide an abstracted view on the object that is being investigated. Models focus only on the aspects of interest to the scientist and hide all other aspects which are deemed irrelevant at that moment. For example, biology is in fact as a whole an abstraction of the underlying physical model of biology that could in principle be completely described by means of quantum mechanics [115]. Even a human being could be perfectly described by the quantum mechanical movement of all its atoms. Yet, even if such a detailed quantum mechanical model could be computed, it would not provide answers to higher-level questions like *'How can we cure a person who suffers from cystic fibrosis?'*[3]. However, an abstracted biological model that only incorporates the relevant aspects involved in cystic fibrosis, *does* provide insight in for example which genes are important in this disease.

A second reason for modeling is that an abstraction of reality often leads to less complex models with less variables to be determined. This means fewer data requirements to uniquely determine the value of each of the variables.

[3] *Cystic fibrosis* or *mucoviscidosis* is a hereditary disease that affects mainly the lungs and digestive system.

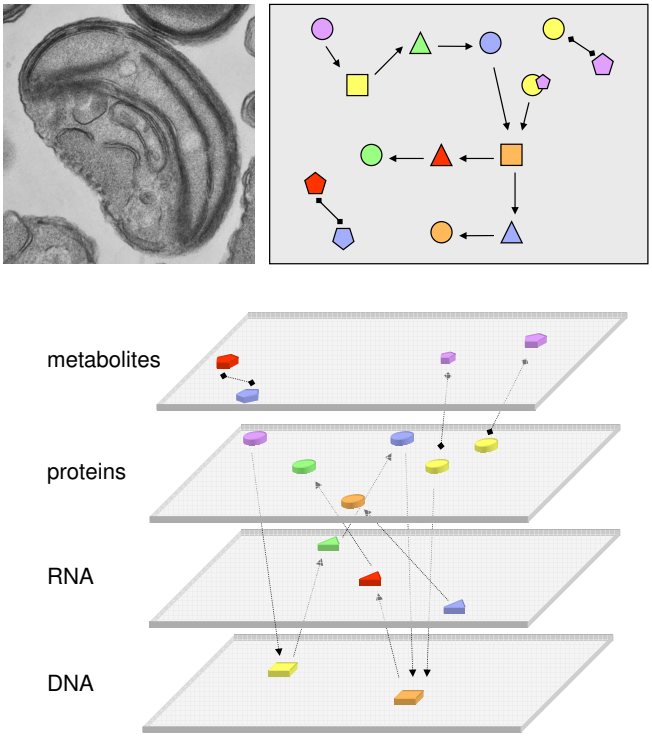


Figure 1.11: ***Different views on the regulatory mechanisms in a biological cell.*** ***(a)*** *Image of a eucaryotic cell.* ***(b)*** *Schematic diagram of the interactions between the different entities (DNA, RNA, proteins, metabolites, ...) in a cell.* ***(c)*** *These regulatory interactions can be viewed from a number of complementary angles and each of these angles has a number of different technologies for performing measurements on the entities it describes. For example: DNA sequencing using polymerase chain reaction (PCR) at the DNA level; gene expression microarrays at the RNA level; 2D gel electrophoresis and mass spectrometry at the protein level; and chromatography, nuclear magnetic resonance at the metabolome level.*

Moreover, the process of determining these values for a model is more likely to be computationally tractable for less complex models.

Once such a model is obtained from the available data, there is a need for *model validation* by comparing the model with the real object. In domains like molecular biology the real object, for example the gene regulatory network of an organism, is at best only partially known. In such circumstances, it becomes useful to validate a model using *simulated* data from a known computational model rather than using measured data on the real but unknown organism.

In the first part of this thesis, a novel model is proposed to simulate large gene

regulatory networks and to generate simulated gene expression data from it. We will show in Chapter 3 that such simulated models of transcriptional regulation reveal properties of inference algorithms that are difficult or even impossible to infer using biological data only. The second part of the thesis focuses on the description of an abstracted model of transcriptional regulation, namely by means of a biclustering model. We propose a probabilistic approach to identify overlapping regulatory modules using the Probabilistic Relational Model framework [55, 61, 94].

1.6 Organization of the thesis

An overview of the thesis structure is given in Figure 1.12. In this Chapter, an introduction was given to the domain of molecular biology. The main microarray techniques were discussed and how the introduction of these high throughput techniques has led to the domain of *systems biology*. In Chapter 2, a broad overview is given of the current state of the art regulatory network inference methods. Chapter 2 helps in understanding the need for an integrative modeling approach to combine the heterogeneous omics data in an abstracted model of a cell.

The next chapters of this thesis are divided into two parts, each highlighting an aspect of the needs that are formulated in Chapter 2. The first part, Chapter 3, discusses the benefits of using proper synthetic benchmarking data to assess network inference algorithms by introducing SynTReN, a framework for generating data from synthetic transcriptional regulatory networks. The operational characteristics of three well known network inference algorithms are determined. The results show that synthetic data provides additional information about the operational characteristics of inference algorithms that is difficult or impossible to obtain by means of biological data only.

The second part, Chapters 4 and 5, focuses on relational data mining and more specifically on the *Pro*Bic biclustering model for the identification of transcriptional regulatory modules. Chapter 4 provides a brief overview of relational data mining and introduces Probabilistic Relational Models (PRMs). The theory and notation of PRMs is explained by means of a fictitious illustrative example on a hospital database containing data of Influenza infections.

Chapter 5 introduces a novel biclustering model, *Pro*Bic, that is based on this PRM framework. The model choices and the algorithmic strategies to make the model computationally tractable are explained in detail. An evaluation on both synthetic and biological data illustrates the strengths of the model in identifying overlapping biclusters in both synthetic and real biological data.

Finally, Chapter 6 summarizes the main research results of our work and proposes an outlook for future research in this domain.

1.7 Achievements

Part I:

- We designed a gene network generator and simulator that is able to simulate large regulatory networks with thousands of genes. Current state-of-the-art dynamic simulators that simulate networks up to maximally a few hundred genes. By reducing the data generation to only steady-state solutions, the simulation of a network comprising thousands of genes becomes computationally tractable.

- Our results show that the choice of network topology for the simulated data can profoundly influence the quality of the results of some inference algorithms. While inference algorithms are often tested on simulated data, the topology of the underlying network is usually not considered as key factor. Disregarding this aspect leads to biased and possibly faulty conclusions based on the results on simulated data.

- Different inference algorithms were applied to simulated datasets with various characteristics. The results show a qualitatively very different response of the algorithms with respect to parameters of the simulated data such as noise, amount of data and interaction types. These results also prove that simulated data is useful to provide more insights in the operational characteristics of an algorithm that are complementary to the insights gained from experiments on real biological data and that are unlikely to be discovered by means of biological data only.

Part II:

- An efficient biclustering algorithm, called *Pro*Bic, has been developed within the framework of probabilistic relational models, requiring no prior discretization of the gene expression data.

- The biclustering model naturally deals with missing values and noise due to its probabilistic nature. This leads to robust identification of biclusters under various settings of noise and missing values.

- Both global and query-driven biclustering can be combined within a single model-based approach and the query-driven biclustering has been proven robust with respect to outlier genes in the set of *seed genes*.

- *Pro*Bic simultaneously identifies multiple overlapping biclusters and an extension to *Pro*Bic allows to group both correlated and anticorrelated genes within a single bicluster.

- The powerful combination of PRMs with an Expectation-Maximization approach allows *Pro*Bic to be easily extended to incorporate additional

data sources, ultimately leading to the identification of regulatory modules with associated condition annotation, regulatory motifs and transcription factors.

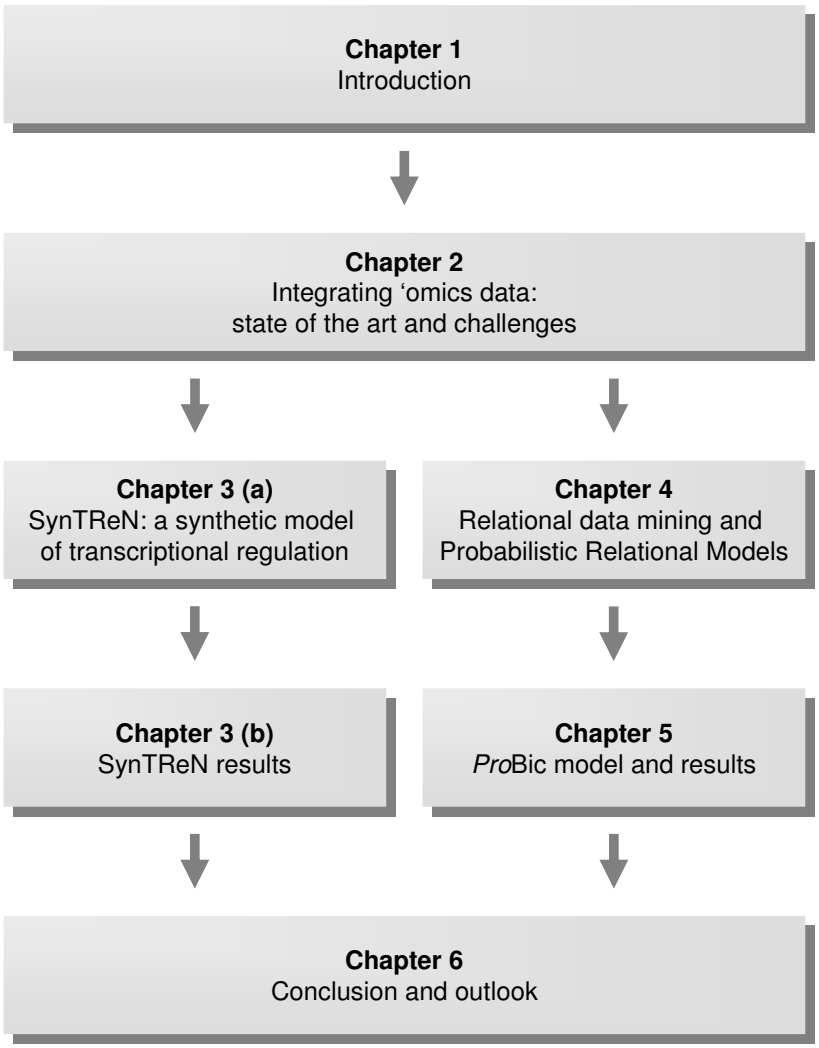


Figure 1.12: ***Organization of the thesis.***

Chapter 2

Integrating *omics* data: state of the art and challenges

2.1 Introduction

With the introduction of microarray technology [95, 140] experiments, a number of new techniques were introduced to analyze gene expression data obtained from these experiments. In this chapter, we first give an overview of studies that describe the reconstruction of transcriptional networks solely from mRNA expression data. Traditional methods for network inference from gene expression data consider every gene as an individual node in the network, and their goal is to infer all interactions between these nodes (Figure 2.1a). Because of the large search space when treating all genes as individual nodes, most of these methods have extensive data requirements obviating their practical usage. Many of these methods generate an ensemble of possible solutions and further postprocessing of these results is often required. However, for a biologist, the primary interest does not lie so much in reconstructing interactions between all genes but in recovering the interactions between the main mediators of the signal transduction, being the regulators and their target genes. Conceptual simplifications that reduce the complexity of the inference problem are therefore possible [159].

Historically, a first category of techniques that made abstraction of the underlying regulatory network was focused on the identification of genes that are significantly over- or under-expressed under the tested experimental condition(s) [10]. A second category of techniques focused on clustering genes that exhibited a similar *expression profile* over all the conditions. Gene clustering approaches are until today still one of the main methods used by biologists to gain insight in which genes are correlated to their genes and/or pathways of

interest. In 2001, Cheng and Church introduced *biclustering*: the simultaneous clustering of both genes and conditions in gene expression data analysis [29]. Since then a variety of different biclustering algorithms have been developed (see [110]) with different measures for identifying good biclusters.

Recently, there is also a growing interest in the modular description of regulatory networks [70, 79]. Genes that are coexpressed in a subset of conditions and that perform similar interactions within the regulatory network, can be grouped into a *regulatory module* [79]. The genes in such a *regulatory module* share a similar expression profile for either a subset or the complete set of conditions and also have a number of other properties in common such as a common set of regulators or a common GO annotation.

Using a module representation, all genes within a module can be described by the same set of parameters instead of using an individual set of parameters per gene (illustrated in Figure 2.1b). This reduction in number of parameters is not only useful for reducing the model complexity but it also provides additional insights in the structure and organization of the regulatory interactions between the genes. We will discuss network reconstruction methods that are based on this simplified network representation in Section 2.3.

With the availability of heterogeneous omics data, the problem of network or module inference becomes even more tractable. Different omics data unveil distinct and often complementary aspects of regulatory networks and their integration allows a more complete insight into the regulatory networks. Here, we will focus on how well distinct computational methods for inference of transcriptional networks can deal with the specific biological features of relevant high-throughput data. It should be noted that the methods described throughout this Chapter are not organism specific although most of them have been field-tested on *S. cerevisiae*, being the most extensively studied model organism [28].

2.2 Reconstruction of transcriptional networks using gene expression data

Gene expression data used for network inference can either be static or dynamic. Static experiments measure gene expression after the cell has adapted to its new environment, for instance if the cell or pathway under study has reached a steady state. Dynamic experiments on the other hand profile the changes in expression level during cellular adaptation. While dynamic experiments inherently contain much more information on the causal interactions between the genes, we will however focus on algorithms that rely on static expression data: most publicly available microarray data is static and moreover, static algorithms can often also deal with dynamic expression data by treating

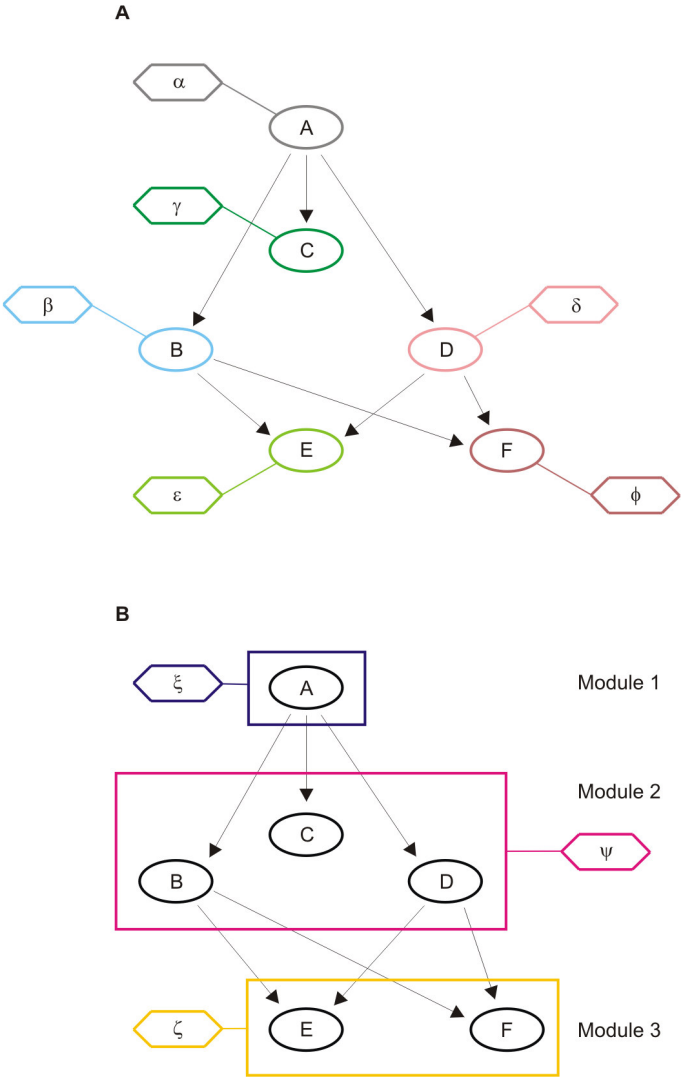


Figure 2.1: ***Graphical representation of networks.*** ***(a)*** *A complete network: each node corresponds to a single gene and is represented by a colored oval. The arrows correspond to the interactions between the genes. For each gene, a unique set of parameters (indicated by hexagons and a Greek letter) describes how the expression of that gene depends on the expression levels of its parents.* ***(b)*** *A module network: each node corresponds to a single gene, denoted as black ovals. The arrows correspond to the interactions between the genes. Genes that depend on the same parents are grouped into modules. For each module, the module parameters (indicated by Greek letters) describe how the expression of all genes within the module depends on the module's parents. A single set of parameters is thus shared by all genes in the module (groups indicated by squares).*

it as static.

An important and often underestimated issue is the preprocessing of the gene expression data prior to the inference of networks. Although microarray technology produces continuous data, many methods require data discretization prior to further analysis. Data discretization implicitly assumes that in a large compendium of microarrays the complete dynamic range of expression values was observed for each gene. This complete range can then for example be subdivided into discrete levels such as high, basal, or low expression level. This discretization step is critical due to the potential loss of information. Also interpreting discretization levels as over-, basal and under-expression should be treated with caution, because observing the complete dynamic range of a gene can never be guaranteed unless a large compendium of data is used. This problem is exacerbated as expression values are often expressed relative to a reference (i.e. when using two-color based array techniques [103]). Large compendia consist of a concatenation of separately performed array experiments that rarely use the same reference [59]. Interpretation of what is over- or under-expressed should always be related to the proper reference.

The classical approaches for network reconstruction from gene expression data aimed at inferring the interactions between all genes. Methods based on Boolean models, Bayesian networks, differential equations and hybrids of those have been described (for exhaustive overviews we refer to D'Haeseleer et al. [45], van Someren et al. [163] and de Jong et al. [38]). Although some of these methods have lead to biologically relevant findings, in general the size of the currently available gene expression data sets does not meet the extensive data requirements for most of these methods. The number of experimental data points is still much smaller than the number of parameters to be estimated. This problem of under-determination is aggravated by the low signal to noise level of microarray data [167] and the inherent stochasticity of biological systems [51, 133]. Therefore, inferring transcriptional networks using the methods described below is usually restricted to small networks or to situations where much data is available.

2.3 From networks to modules

However, there is recently a major interest in the identification of module networks. Reformulating the problem of inferring networks as a problem of inferring module networks can greatly simplify the complexity of the problem. We adopt the terminology introduced by Segal et al. [145] for modules and regulatory programs: a set of genes, coexpressed under all or under a particular set of conditions, is assumed to undergo similar interactions within the network. Such a gene set is called a *module*. A *regulatory program* is defined as the set of regulators of which the concerted action is responsible for the condition

dependent expression of the genes in the corresponding module. The module network inference problem consists of two subtasks, namely identification of the modules and identification of the regulatory programs.

As module networks are a conceptual abstraction of the real networks, some biological considerations have to be made. Module networks are condition dependent by definition, meaning that the genes of a module are only co-regulated under a specific subset of conditions (i.e. tissues, time points, environmental conditions, etc.). In order to grasp this context specificity of a module, searching for modules in a large compendium of gene expression data not only implies identifying sets of coexpressed genes, but also selecting the conditions in which the genes exhibit a correlated behavior. This context specificity is reflected in the combinatorial composition of the regulatory program. Due to the combined action of regulators, genes of a module behave similar in a condition dependent way [105]. This is illustrated by a hypothetical example in Figure 2.2 (the hypothetical example is a generalization of our own observations and those described by Ihmels et al. [78]).

A gene expression dataset can therefore be subdivided in several overlapping context dependent modules. Modules comprising many conditions can be expected to contain few genes (called hereafter seed genes) with a potentially highly related function. Indeed, the more conditions genes appear to be co-expressed in, the more similar their regulatory program tends to be and the more connected their role in the pathway becomes. In a module, the number of genes will usually increase with a decreasing number of conditions. Obviously, there will be more genes that only share part of their regulatory program, i.e. the part that is active under the tested set of conditions. The fewer the number of conditions included in the module one considers, the less stringent the requirements on the overlap in the regulatory program becomes (Figure 2.2). Although these considerations seem trivial from a biological point of view, these properties of modules and programs make inferring modules and their corresponding regulatory program a non-straightforward task.

2.4 Identification of modules using gene expression data

Biclustering algorithms are well suited to identify these regulatory modules. They assign genes to condition dependent and potentially overlapping regulatory units, i.e. modules. In contrast to classical two-way clustering approaches [26], these biclustering algorithms do not group genes and conditions independently, but simultaneously, thereby identifying subsets of genes that are each correlated under a subset of conditions [79]. For the purpose of clustering, microarray experiments are usually organized in an expression matrix, where the rows correspond to genes and the columns correspond to different condi-

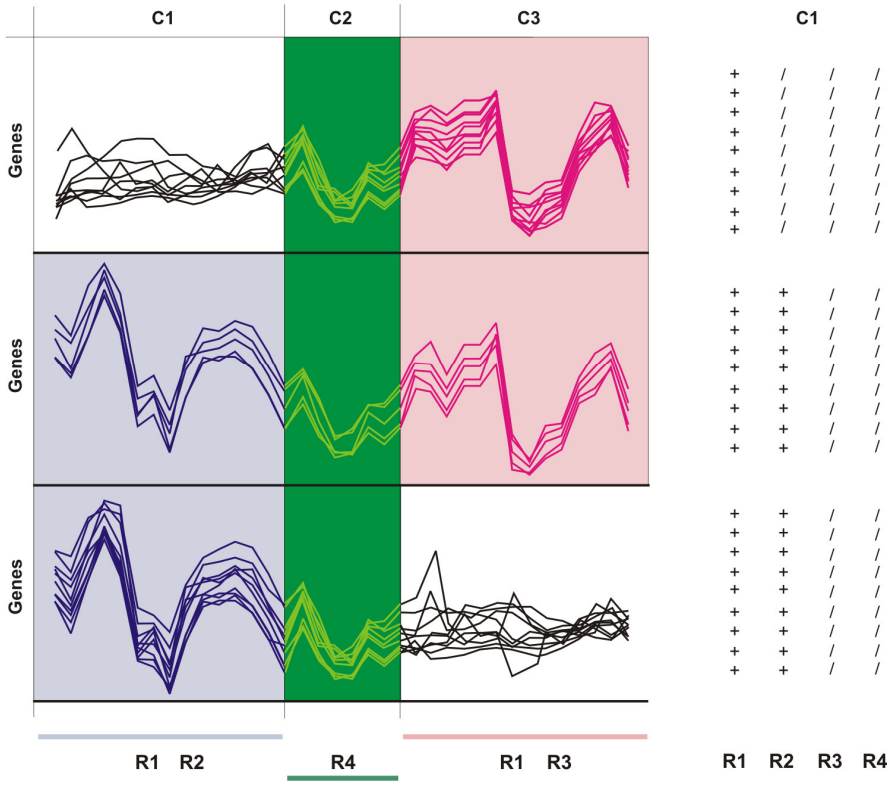


Figure 2.2: ***Hypothetical example of higher order module organization.*** ***(a)*** *C1, C2 and C3 represent three unrelated condition sets. the R*i *represent different regulators active in the respective condition dependent regulatory programs: R1, R2 are active in C1; R1, R3 are active in C3; R4 is active in C2. Distinct partially overlapping modules exist. Modules consisting of a few genes, tightly coexpressed in many conditions can be hypothesized to be associated with a highly specific function (horizontal middle panel). They consist of genes that respond to the same regulatory program and are coexpressed under all conditions. As modules are extended with more genes, the number of conditions can be expected to decrease. Genes within such extended modules only share part of the regulatory program, i.e. the one that is active under the selected conditions (top and bottom panel). Which of these overlapping modules will be detected by biclustering depends on the specificities of the algorithm and the parameter choices when applying the algorithm.* ***(b)*** *Hypothetical ChIP-chip result for the respective regulators obtained from a ChIP-chip compendium measured in C1 only. +: binding between target gene and regulator is observed; /: no binding is observed. Since only ChIP-chip data of condition C1 exists, the data contains much missing and conflicting information when extrapolating it to other conditions. Therefore, such information should be interpreted with caution for data-integration.*

tions. Biclustering algorithms can be grouped according to different criteria such as whether they are based on probabilistic methods or not, whether they allow for overlapping modules or not, whether they search for all modules at once or try to identify the modules separately in subsequent runs, whether the obtained modules are self-consistent or not, and whether they use seeds as a starting point for module identification or not. From a biological perspective, having an algorithm that allows for overlapping modules is desirable (see Figure 2.2 and the considerations made above). From an algorithmic perspective, self-consistency of the module (a criterion introduced by Ihmels et al. [78]) allows for generating optimal and potentially overlapping modules instead of optimizing the global data partitioning. The use of seeds, defined as initial sets of genes around which a module is formed, leads to a straightforward extension for data integration (see further). The different module inference algorithms together with their most important properties are summarized in Table 2.1 and a comparison table of the main biclustering algorithms is shown in Table 5.1.

2.5 Simultaneous identification of modules and their regulatory program

Biologists are not only interested in inferring the module composition but also in reconstructing the regulatory program. Because the regulatory program determines the behavior of the genes in the module and the presence of a module reveals which programs are active, it does make sense to simultaneously infer the active programs and to partition the genes into modules. A first step into this direction is the module networks method developed by Segal et al. [145], which is inspired by the probabilistic relational model (PRM) framework [55, 61, 62, 94] (see Figure 2.2).

While Friedman et al. [54], in the initial applications of probabilistic models for network inference, assigned each gene as a separate node with its own parameters to the Bayesian network, Segal et al. [145] grouped genes into modules, where genes belonging to the same module share the same parameters and have the same set of regulators. This considerably reduces the number of model parameters to be estimated and at the same time increases the number of data points available for estimating each parameter. For each module, the effect of the set of regulators on the expression profile of the genes in the module is modeled as a transcriptional program by using a regression tree. The iterative procedure uses an Expectation-Maximization (EM) algorithm to search the optimal regulatory program for each module (M-step, using a regression tree for the regulatory program) and to subsequently reassign each gene to the module of which the program best predicts its behavior (E-step, using the model score). Although very innovative, the approach still has some

Study	Method	Data types	Overl. mod.	Mod. netw.	Subset cond.	Integr. add.	WWW
Eisen et al. [49]	Pairwise clustering	gene expression	no	no	no	no	http://genome-www.stanford.edu/clustering/
Ihmels et al. [78]	Signature Algorithm based	gene expression	yes	no	yes	yes	http://www.weizmann.ac.il/home/barkai/modules/
Tanay et al. [154]	Weighted bipartite graph	gene expression, protein interaction, growth phenotype, TF binding	yes	yes	yes	yes	http://www.cs.tau.ac.il/~rshamir/samba
Gash and Eisen [58]	Modified k-means	gene expression	yes	no	no	no	http://rana.lbl.gov/FuzzyK
Sheng et al. [150]	Gibbs sampling, Bayesian network	gene expression	yes	no	yes	yes	http://www.esat.kuleuven.ac.be/~qsheng/query_driven.html
Segal et al. [145]	PRM-inspired	gene expression	no	yes	no	yes	http://ai.stanford.edu/~erans/module_nets/
Cheng and Church [29]	biclustering	gene expression	no	no	yes	no	http://arep.med.harvard.edu/biclustering/
Getz et al. [63]	CTWC	gene expression	no	no	yes	no	http://www.weizmann.ac.il/physics/complex/compphys/
Kluger et al. [91]	Spectral biclustering	gene expression	no	no	yes	no	
Lee et al. [99]	Bayesian network, information theory	gene expression, biological annot.	yes	yes	no	yes	http://biosoft.kaist.ac.kr/~dhlee/monet/index.html
De Bie et al. [37]	Apriori algorithm inspired, statistics	gene expression ChIP-chip, sequence data	yes	no	no	yes	http://www.esat.kuleuven.ac.be/~kmarchal/Supplementary_Info_PSB2005/SuppWebsiteYeastPSB.html
Bar-Joseph et al. [12]	Statistics	gene expression, ChIP-chip	yes	no	no	yes	http://psrg.lcs.mit.edu/GRAM/Index.html
Lazzeroni and Owen [98]	Two sided cluster analysis	gene expression	no	no	yes	no	http://www-stat.stanford.edu/~owen/plaid/
Gu et al. [64]	Bayesian	gene expression	no	no	yes	no	http://www.people.fas.harvard.edu/~junliu/BBC
Lemmens et al. [102]	Itemset mining	gene expression, sequence data	yes	no	yes	yes	http://homes.esat.kuleuven.be/~kmarchal/Supplementary_Information_Lemmens_2008/Index.html

Table 2.1: Columns in the table denote the different attributes for which the methods were compared. Method: *the basic methodology used,* Data types: *the different data types which the algorithm combines in the study;* Overl. mod.: *indicates whether the method is able to generate overlapping modules;* Mod. netw.: *indicates if the method generates a module network (module and transcriptional program);* Subset cond.: *indicates if the modules are defined for a subset of the conditions or for all conditions;* Integr. add.: *indicates whether it is possible to easily extend the algorithm to incorporate additional data sources other than the data sources described in the study;* WWW: *a link to the online resources/software of the algorithm.*

shortcomings, mainly since it uses only gene expression data to construct the regulatory program.

In the original setup of Segal et al. [145], a large set of candidate regulators is selected based on their annotation, while the regulatory program of each module is composed of the subset of candidate regulators for which the expression profiles best explains the expression profile of the genes in the module. This criterion complicates distinguishing between regulators that are actually causing the modules behavior (and thus belong to the regulatory program) and those for which the observed expression behavior is a consequence of the action of the program (and thus belongs to the module). Moreover, constitutively expressed regulators activated post-transcriptionally, will never be part of the regulatory program because their expression profile will not correlate with the genes in the module. The number of regulators for which the expression profile correlates well with the profiles of its target genes, is limited anyhow as Herrgard et al. [72] showed that in yeast over 80% of the tested pairs of expression profiles between regulators and targets were not significantly correlated. In the method of Segal et al. [145], context specificity of the modules is not explicitly taken into account (no conceptual biclustering) because genes belonging to a module are required to be coexpressed over all conditions tested. By definition, a gene can only belong to a single module and overlapping modules are therefore not possible. Segal et al. [145] applied their method to the Gash et al. dataset [59], a large scale microarray experiment (173 arrays) assessing expression changes under various stress conditions in the yeast *S. cerevisiae*, and identified 50 modules involved in various processes. They proved the biological potential of their method by experimentally validating three of the hypotheses that followed from their predictions.

The model of Segal et al. has been further extended by Michoel et al. [116] who describe an alternative and faster approach for learning regulatory programs. In Joshi et al. [85], an ensemble method is used to identify more coherent modules which further improved the module learning approach.

2.6 Network inference using data integration

Among the approaches developed to infer complete networks, in particular probabilistic approaches have been extended to integrate heterogeneous data sources. Additional data is used to supplement the expression data for example by using ChIP-chip data or protein interaction data as priors for Bayesian networks [20, 69, 68, 80, 99, 153]. Biclustering algorithms that start from a set of seed genes to define biclusters (see Table 2.1) can, to some extent, integrate heterogeneous data: the seeds can be defined by using other data sources and can thus be considered as prior information to the biclustering algorithm [79].

An example of a deterministic method for data-integration is the MA-networker

algorithm for integrative modeling of expression and ChIP-chip data [56]. This algorithm uses multivariate regression of mRNA expression levels on the genome-wide binding profiles of a large number of transcriptional factors (ChIP-chip data) to explain to what extent each transcription factor is responsible for the observed changes in mRNA expression in a single microarray experiment. When performing the regression procedure in parallel on a compendium of multiple microarray experiments, a transcriptional factor activity profile is obtained that implicitly expresses the conditional dependence of a specific regulator on the tested experimental conditions, indicating whether the regulator is responsible for the expression changes per experimental condition. For the identification of additional target genes of a specific regulator, the method of Gao et al. [56] is original in that, in contrast to most other studies, it does not search for genes of which the expression profile is highly correlated with the one of the regulator but for genes with an expression profile highly correlated with the activity profile of the regulator (defined as the coupling strength). Although the method searches for the condition dependent activation of a gene by one regulator it does not yet use this information to determine the concerted action of more regulators, i.e. to compile complete regulatory programs [56]. MA-networker was applied to the yeast ChIP-chip data of Lee et al. [101] and a compendium of 750 microarray experiments covering different physiological conditions. They found that 58% of the genes whose promoter was bound by a regulator were true targets and that a set of target genes of which the expression profile exhibits a large coupling strength with the activity profile of a specific regulator was significantly enriched for specific functional categories.

The following methods aim at identifying modules and regulatory programs. Similar to Gao et al. [56], they treat additional data sources with equal importance compared to gene expression data (contrary to other approaches where additional data sources are treated as prior).

Wang et al. [164] propose a heuristic semi-integrative, semi-sequential method to combine motif- and gene expression data in order to search for target genes of a particular transcription factor, the context specificity of the transcription factor, and the combinatorial control between different regulators. Their method is based on the assumption that if a transcription factor is activated under a particular condition, its target genes should have similar responses as those observed in a perturbation experiment of that transcription factor. Regulatory motifs recognized by a particular transcription factor, and their corresponding targets, are identified with the REDUCER algorithm [52] in an experiment where the particular transcription factor is perturbed (e.g. overexpressed, mutated). Based on this information, a score vector for the perturbation experiment is constructed which consists, for each potential target gene, of a value that increases with the ratio of overexpression of that gene in the prevailing experiment and with the number of motifs for the transcription factor of interest. Besides for the perturbation experiment, this vector is also calculated for

different other microarray experiments. From the correlation between these calculated score vectors, the conditional activity of the transcription factor is derived. Since in some microarray experiments more regulators can be active simultaneously, the overlap in their target genes is used to derive combinatorial action of different transcription factors. Based on a microarray compendium (which comprised the Gash et al. [59] and Spellman et al. [152] datasets), for 28 transcription factors of which a perturbation experiment was available (from the Rosetta compendium [76]), Wang et al. [164] identified the corresponding target genes, motifs and relevant conditions.

Harbison et al. [66] and Kato et al. [86] use a heuristic approach that is partially integrative. For each regulator, they first compile reliable lists of target genes, based on the integrated knowledge from literature [101, 66], ChIP-chip [66, 86], and comparative genomics data [66]. Subsequently, the search for statistically overrepresented motifs in the promoter regions of these target genes results in the identification of the motif tags characteristic for each of the regulators. Kato et al. [86] go one step further in reconstructing the modules and programs by searching for statistically overrepresented motif combinations. Genes of which the promoters contain a particular motif combination and that share a similar expression profile over time comprise a module. In a final step, regulatory programs are identified based on the ChIP-chip data by determining the identities of the regulators that are statistically overrepresented in the genes of the respective modules. When applying their method to the ChIP-chip data of Lee et al. [101], they specifically focused on the cell cycle and could identify most of the previously described cell cycle transcriptional complexes.

A conceptual extension to the previously mentioned heuristic methods is proposed by Bar-Joseph et al. [12] and De Bie et al. [37]. Regulatory programs and module seeds are defined in a joint learning step based on ChIP-chip and gene expression data [12], or based on ChIP-chip data, gene expression data, and motif data [37]. In the former approach (*GRAM*), seeds are defined by identifying sets of genes with a common set of transcription factors and having a highly correlated expression profile (determined by microarray analysis). In the latter approach (*ReMoDiscovery*), module seeds are maximal gene sets of which the expression profiles are highly similar and that have a minimal set of regulators and motifs in common [37]. The shared set of seed regulators and motifs corresponds to the regulatory program determining the observed coexpressed behavior of the module seed genes. Searching for maximal gene sets that meet these requirements on all three datasets translates into a combinatorial problem, which is solved by a modification of the *Apriori* algorithm [37]. Because the initial seed discovery in both approaches relies on stringent criteria (information in all datasets has to be mutually consistent), the seed modules are likely to underestimate the true module size. For this reason, both algorithms use a second module extension step. Bar-Joseph et al. [12] extend the module seeds by first identifying candidate genes with an expression profile sufficiently similar to the seed profile and with a sufficiently low

p-value for the binding of each of the individual regulators of the module seed. A combined P-value based on the individual p-values for all module regulators is calculated for each of the candidate genes passing these requirements and the gene is added to the module if the combined P-value is sufficiently low. ReMoDiscovery contains a second module extension step where additional genes are identified for which the expression profile is highly correlated with that of the seed genes. The optimal size in number of genes of the module is determined by the correlation coefficient resulting in a module with the largest enrichment in seed motifs and regulators. The regulatory modules detected by both approaches can be used as input sets for motif detection tools. Note that both approaches [12, 37] yield few false positive modules, but they fail to identify modules if not all data sources separately confirm the presence of a seed module. Neither Bar-Joseph et al. [12] nor De Bie et al. [37] in its original implementation explicitly take into account the conditional nature of the regulatory program. De Bie et al. [37] solve the problem by grouping microarray experiments performed in the same experimental condition and applying the algorithm to each group of microarrays separately. Only when the regulatory program is active in the specific dataset, the seed module can be extended. Both methods were applied to the yeast ChIP-chip compendia and various microarray experiments in yeast. Although using slightly different datasets, both groups identified a similar number of modules, involving a comparable number of regulators. By performing gene specific ChIP-chip experiments, Bar-Joseph et al. [12] experimentally validated a random selection of predictions proving the potential of their approach.

Lemmens et al. [102] further extend the module discovery approach of *ReMoDiscovery* by not only searching for sets of highly co-expressed genes that share controlling regulators, but by also selecting the experimental conditions for which the selected genes are coexpressed. In this algorithm, called DISTILLER, genes are no longer required to be co-expressed over all conditions. The framework applies advanced itemset mining approaches to efficiently search the complete space of possible modules.

Xu et al. [172] extended the module networks framework of Segal et al. [145] (see higher), by incorporating ChIP-chip data. For identifying the most likely candidates of a regulatory program, they select regulators for which the expression profile shows a high mutual information with the one of the module genes (comparable to the approach of Segal et al. [145]) and regulators with high binding probabilities based on ChIP-chip data. The binding probability between a regulator and its target genes is also regarded as a structure prior to the Bayesian score, which scores the inferred module networks. As a result, the score of the resulting network is increased both when there is a high correlation between the expression profile of the regulator and the module genes (when calculating the regression tree that derives the regulatory program) and when the binding probability between a regulator and its targets is high (in the form of a structure prior). This joint scoring allows different weaker indications

from separate data sources to be joined to a significant indication, to indicate for example that a gene is part of a module. It also allows constitutively expressed regulators with low location probability to be part of the regulatory program. As with the method of Segal et al. [145], conditional dependence of the regulatory programs is not taken into account. Using the ChIP-chip compendium of Lee et al. [101] and the microarray experiments of Gash et al. [59] and Spellman et al. [152], Xu et al. [172] identified 50 modules involving 86 regulators covering a wide range of cellular/physiological processes.

Another method for module detection is SAMBA, developed by Tanay et al. [154], which uses a bipartite graph based representation of the data where one subset of nodes represents genes and the other subset represents the properties derived from the distinct heterogeneous data. An edge represents the assignment of a property to a gene with the weight of the edge being indicative of the statistical strength of the assignment. The problem is then reduced to finding 'heavy' subgraphs in a weighted bipartite graph. A graph-based biclustering algorithm [155] is used to identify modules (i.e. a set of genes that show similarities only in a subset of properties). Like for other biclustering algorithms, overlapping modules are allowed. This method thus fully exploits all data sources, allows dependency of the program, not only conditioned on the expression data but also on the other data sources. This is important, considering that ChIP-chip or protein interaction data, assessed under specific conditions might not be supported by, for instance, expression data measured under different conditions. One drawback of this method, from a biological point of view, is that while the uniform representation of the heterogeneous data allows the automatic identification of modules, the compilation of the regulatory programs is not automatically derived from the analyses. It is also unclear how the different sets of properties of the genes should be weighted compared to one another, while this will have a profound impact on the identified modules. Integration of additional data sources is straightforward as long as the data can be described as gene properties. Tanay et al. [156] identified 1200 significant modules in a large yeast compendium of heterogeneous datasets. 86% of the modules were based on more than one dataset and for the construction of 68% of the modules at least three different data sources were used, indicating the importance of using complementary information.

Besides the data integration methods mentioned above, there are many other methods that focus on different aspects of regulation, such as the combined identification of regulatory motifs and coexpressed genes (e.g. [27]).

2.7 Assessment and validation of inference algorithms

A thorough validation of the results of network inference algorithms can be challenging. One of the main validation strategies is to compare the predicted interactions with those that have been previously described in literature and which were independently verified by means of wet-lab experiments. Even for model organisms such as *Escherichia coli* and *Saccharomyces cerevisiae*, the set of interactions is only partially known. And since not all interactions are known, a dataset with known non-interactions is not available. Therefore, this validation strategy does not take into account *false positives*: interactions that are predicted by the algorithm but that are not described in literature. Such predictions can either be novel interactions or wrong predictions of the algorithm. To address this problem, some authors have artificially constructed datasets with 'known non-interactions', e.g. by assuming that proteins that are expressed in different cellular locations do not interact or by assuming that genes with different gene ontology categories are unlikely to interact. While these datasets provide a partial answer to the stated problem, the dataset with 'known non-interactions' is highly biased towards a particular set of interactions. The results of an assessment using such negative datasets should therefore be interpreted with caution. A second widely used validation strategy is to indirectly measure the quality of the results using enrichment scores, e.g. functional enrichment using gene ontology (GO) annotation. The underlying assumption in this approach is that interacting genes are more likely to have the same GO annotation than non-interacting genes. This approach is mostly used for clustering and partitioning algorithms. While this approach uses independent datasets, it is only an indirect indication of the quality of the results. Finally, wet lab experiments provide the most reliable validation of predicted regulatory interactions. The main disadvantages of wet lab experiments are its expensiveness in terms of price and time.

Apart from validating the results of inference algorithms, researchers are also interested in gaining statistical knowledge on their algorithm: *'What is the performance for increasingly noisy data?'*, *'What is the most optimal parameter setting?'*, *'How robust are the results?'*, etc. This requires repeatedly testing them on large, high-quality data sets obtained from many experimental conditions. Unfortunately, such experimental datasets are usually not available and moreover, the true underlying network of interactions is at best only partially known. Due to these limitations of real experimental data, the use of simulated data as an additional tool for benchmarking structure learning algorithms is gaining interest [11, 36, 107, 139, 175]. Several models have already been proposed for this purpose, including Boolean [4, 134], continuous [77, 114, 176] and probabilistic [114] approaches. Most current network simulators [77, 92, 114, 151, 176] use a set of ordinary differential equations (ODE's). The choice of a numerical so-

lution method, which depends on the desired precision and the specific form of the set of ODE's, can lead to scalability problems. The time complexity of numerically solving a set of ODE's for a given time period, in function of the number of genes, varies between linear and cubic complexity, which makes simulation of large networks computationally difficult.

2.8 Summary

In this Chapter, an overview was given of different network inference algorithms for the reconstruction of transcriptional regulatory networks. An overview is given of inference algorithms that are based purely on gene expression data. The classical approaches for network reconstruction from gene expression data aimed at inferring the interactions between all genes using Boolean models, Bayesian networks and differential equations. Because the number of experimental data points is still much smaller than the number of parameters to be estimated in these methods, the resulting solutions are underdetermined. As a result, there is a growing interest in methods for the identification of module networks, as the number of required parameters is usually much smaller. An overview of these methods was given. Since gene expression data does not provide a complete angle on all aspects of gene regulation, data integration methods have been proposed that infer regulatory networks or module networks using additional data sources such as ChIP-chip or sequence data.

A thorough validation and assessment of the different types of network inference algorithms is difficult due to a lack of large high-quality datasets and the fact that the underlying interactions for these datasets are only partially known at best. The use of simulated data offers an interesting complementary mechanism for gaining statistical knowledge on different characteristics of inference algorithms. As we will see in Chapter 3, this knowledge is unlikely to be discovered by means of biological microarray data only.

Chapter 3

A synthetic model of transcriptional regulation: SynTReN

The results in this Chapter were developed in cooperation with ISLab (Universiteit Antwerpen).

3.1 Background

Developing reliable data analysis methods that infer the complex network of interactions between the various constituents of a living system based on high throughput data, is a major issue in current bioinformatics research [153]. Because data on transcriptional regulation are most accessible, much effort goes to the development of algorithms that infer the structure of transcriptional regulatory networks (TRNs) from this data [45, 48, 54, 119, 127, 146, 157].

Gaining statistical knowledge about the performance of these algorithms, requires repeatedly testing them on large, high-quality data sets obtained from many experimental conditions and derived from different well-characterized networks. Unfortunately, experimental data sets of the appropriate size and design are usually not available. Moreover, knowledge about the underlying biological TRN is often incomplete or unavailable.

As a consequence, validation strategies applied to experimentally obtained data are often limited to confirming previously known interactions in the reconstructed network. However, using such an approach, false positive interactions are for example not penalized. Indeed, assessing the relevance of predicted interactions that have not been experimentally confirmed, is infeasi-

ble. Secondly, the algorithm can usually only be applied to data from a single network, which complicates algorithm design and validation. Due to these limitations of real experimental data, the use of simulated data as an additional tool for benchmarking structure learning algorithms is gaining interest [11, 36, 107, 139, 175].

Throughout this chapter, the term *network generator* is used to denote a system that generates synthetic TRNs and the simulated gene expression data that is derived from these networks. A synthetic TRN consists both of a topology that determines the structure of the network and an interaction model for each of the regulatory interactions between the genes.

Different approaches have been used to create network topologies. The generation of small networks is often based on detailed handcrafted topologies [77, 151, 176]. For producing topologies of large networks comprising thousands of nodes, random graph models have been used [92, 114]. The latter models create graphs that share one or more statistical properties, such as scale-free [166] and small-world [5] properties, with known regulatory networks, in an attempt to approximate biological reality.

For simulation of the regulatory network, the interactions between the genes need to be quantitatively modeled. Several models have been proposed for this purpose, including Boolean [4, 134], continuous [77, 114, 176] and probabilistic [114] approaches. Most current network simulators [77, 92, 114, 151, 176] use a set of ordinary differential equations (ODE's). The choice of a numerical solution method, which depends on the desired precision and the specific form of the set of ODE's, can lead to scalability problems. The time complexity of numerically solving a set of ODE's for a given time period, in function of the number of genes, varies between linear and cubic complexity, which makes simulation of large networks computationally difficult.

We propose ***SynTReN*** (**Syn**thetic **T**ranscriptional **Re**gulatory **N**etworks), a network generator that copes with some of the limitations of previous implementations. Instead of using random graph models, topologies are generated based on previously described source networks, allowing better approximation of the statistical properties of biological networks. The computational cost of our simulation procedure is linear in function of the number of genes, making simulation of large networks possible.

3.2 Model overview

The SynTReN network and data generation process comprises of three essential steps. Figure 3.1 shows the complete execution flow of the process, the shaded area indicates the data generation steps which are executed in SynTReN. The process starts by selecting a network topology that is generated using either a random graph model or by deriving it from a known (biological)

source network. The latter approach is performed by means of two subnetwork selection strategies which will be explained in detail in Section 3.3.3. In the networks that are generated by SynTReN, the nodes represent the genes and the edges correspond to the regulatory interactions at transcriptional level between the genes. In the second step, transition functions and their parameters are assigned to the edges in the network (Section 3.4). In the third step, mRNA expression levels for the genes in the network are simulated under different conditions. After optionally adding noise, a gene expression data set is obtained that represents normalized and scaled microarray measurements.

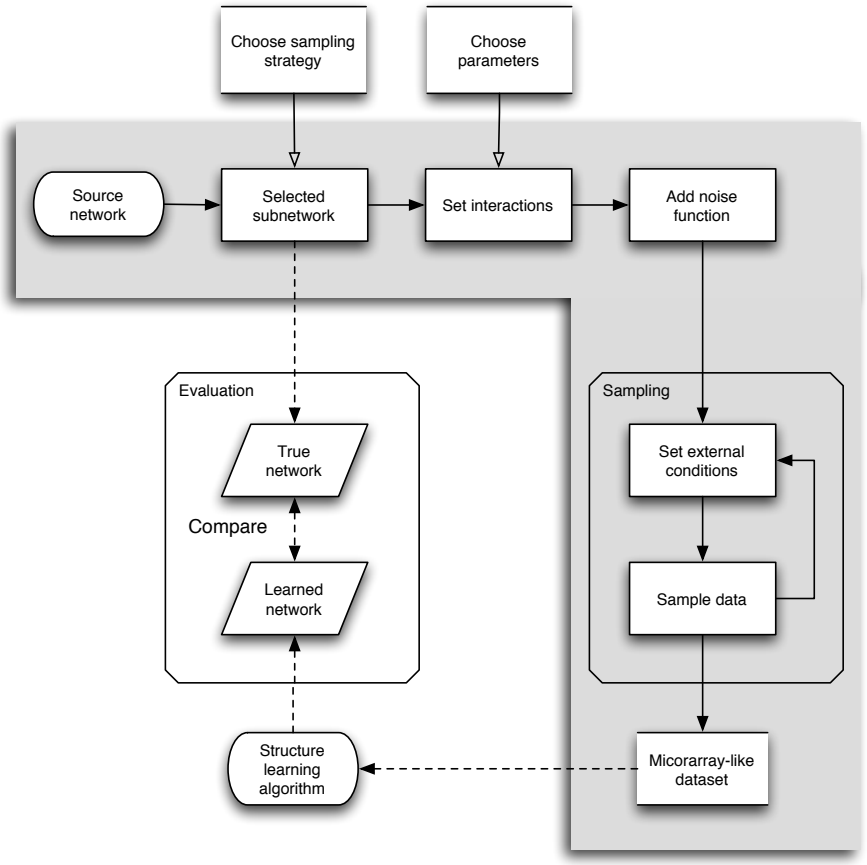


Figure 3.1: ***Overview of the synthetic transcriptional network generator****. The shaded area highlights the data generation steps and the dashed arrows show how the output fits in a validation strategy for a network inference algorithm.*

In order to evaluate to what extent our approach compares to previous approaches in generating networks with topological characteristics of true TRNs, we used well established deterministic and informative measures that can be subdivided in two distinct categories, each of which addresses different aspects of the network structure: high-level (global) measures and low-level (local) measures. The high-level measures, like average clustering coefficient and average path length, depend on knowledge of the complete network structure while the low level measures are derived from local network properties such as for instance the marginal degree distributions [5, 177].

Biological TRNs have specific common structural properties: the small world property [166], indicating a short average path length between any two nodes and the scale-free property [5], indicating the degree distribution of the nodes of the network follows a power law. True biological TRNs also contain specific structural motifs that are statistically overrepresented as compared to random graphs of the same in- and outdegree [118, 149] (e.g. feed forward loop). Synthetic TRNs have been generated using different types of random graph models [92, 114], such as Erdös-Rényi [52], Albert-Barabási [5] and Watts-Strogatz [166] models. These models generate graphs with one or more topological properties observed in biological TRNs. Unlike previous approaches, we generate network topologies by selecting subgraphs from a previously described biological source network (*E. coli* [108, 149] or *S. cerevisiae* [65]).

3.3 Network topology selection

The first step in our setup is to select a network topology of the synthetic transcriptional network. This section provides an overview of the available techniques in SynTReN. We will show that the SynTReN subnetwork selection methods result in topologies that better approximate the characteristics of real biological networks. We will also show that inferring networks from expression data generated with the subnetwork selection methods generally leads to higher quality network inference. Results will be shown that indicate the importance of the network topology choice for a proper assessment of an inference algorithm (Section 3.8.4). In the next sections we will first describe some commonly used random graph models and the SynTReN subnetwork selection methods and then compare the topological characteristics of the networks generated with both approaches.

3.3.1 Random graph models

In [92, 114], different types of random graph models are used to generate a network topology: Erdös-Rényi [52], Albert-Barabási [6] and Watts-Strogatz [166] random graph models. These models and the directed scale free (DSF) model of Bollobás [24] will be briefly described. A more in-depth description

can be found in [6] and [52].

Random graph model (Erdös-Rényi)

An Erdös-Rényi model is one of the most straightforward random graph models. As described in [6], a network that is generated using an Erdös-Rényi model can be defined by a binomial model. The network has a predefined number of nodes N and between every pair of nodes an edge is defined with probability p. Figure 3.2 shows a series of 6-node networks with different p values.

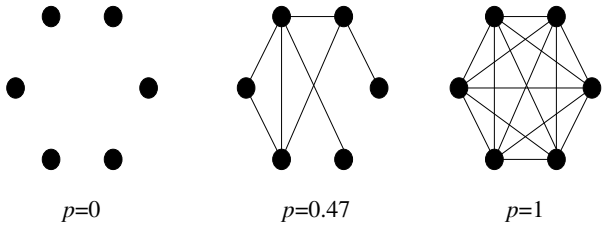


Figure 3.2: ***Erdös-Rényi random networks with 6 nodes, generated with p-values 0, 0.47 and 1.***

Small-world model (Watts-Strogatz)

The small-world model of Watts and Strogatz interpolates between an ordered lattice and a random graph according to the following algorithm (after [6]):

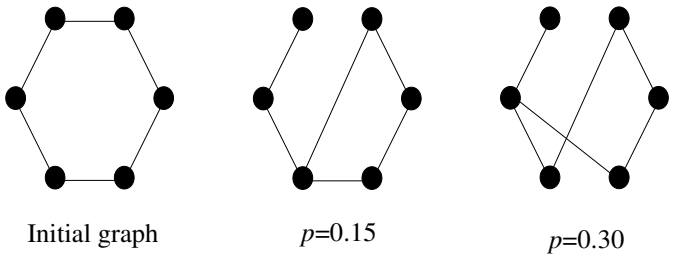
1. Start with a ring lattice with N nodes, where every node is connected to its K nearest neighbors.

2. Randomly rewire each edge with probability p, but avoid self-edges and duplicate edges.

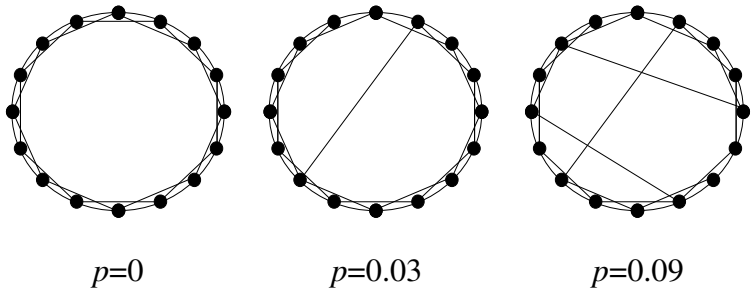
This process introduces long-range connections among the initial short-range connections (see Figures 3.3a and 3.3b), connecting nodes that otherwise would be part of different neighborhoods. These networks have a small-world property: the shortest path between any two nodes is small on average.

Scale-free model (Albert-Barabási)

The model of Albert and Barabási is based on two basic principles: *growth* and *preferential attachment*. The graph starts with a small number (m_0) of nodes, and for a series of time steps, a new node is added each time with a number of edges according to the following rules:



(a) Watts-Strogatz random network with 6 nodes, generated with rewiring probabilities 0 (=initial ring lattice), 0.15 and 0.30 and starting from a ring lattice with two neighbors.



(b) Watts-Strogatz random network with 16 nodes, generated with rewiring probabilities 0 (=initial ring lattice), 0.03 and 0.09 and starting from a ring lattice with four neighbors.

Figure 3.3: ***Example networks, generated using the Watts-Strogatz random graph model.***

1. Growth: at every time step, we add a new node with m ($< m_0$) edges that link the new node to m different nodes already present in the system.
2. Preferential attachment: the probability p that a new node will be connected to an existing node is proportional to the degree of that node.

Based on these rules, it can be proven that the probability that a node has k edges follows a power law [6, 23].

Directed scale-free model (Bollobás)

Bollobás et al. [23] present an extension of the Albert-Barabási model [6] for directed graphs. Following the same basic principles, these graphs grow with preferential attachment depending on in- and out-degrees. The resulting in- and out-degree distributions are again power laws, possibly with different exponents.

The model has five parameters ($\alpha, \beta, \gamma, \delta_{in}$ and δ_{out}) and starts with any fixed initial directed graph G_o. At each time step:

- With probability α, add a new vertex v together with an edge from v to an existing vertex w, where w is chosen according to the probability distribution $d_{in}(w) + \delta_{in}$ where $d_{in}(w)$ is the in-degree of w.
- With probability β, add an edge from an existing vertex v to an existing vertex w, where v and w are chosen independently, v according to $d_{out}(v) + \delta_{out}$, and w according to $d_{in}(w) + \delta_{in}$.
- With probability γ, add a new vertex w and add an edge from an existing vertex v to w, where v is chosen according to $d_{out}(v) + \delta_{out}$.

The following equations hold: $\alpha + \gamma > 0$ and $\alpha + \beta + \gamma = 1$.

3.3.2 Biological subnetwork selection methods

To generate a network topology that resembles a true TRN as closely as possible, network structures are sampled from previously described biological networks. The topologies of the well-described model organisms *E. coli* [108, 149] and *S. cerevisiae* [65] are used as source graphs. Two different strategies to select a connected subgraph from a source graph are implemented.

In a first strategy, called *neighbor addition*, a randomly selected node is chosen as initial seed. Subsequent nodes are added in an iterative process. Only randomly selected nodes that have at least one connection to the current graph, are retained.

In a second strategy, called *cluster addition*, a randomly selected node and all of its neighbors are selected as initial graph. In each iteration, a randomly selected node and all of its neighbors are added to the graph. Similarly, only nodes that have at least one connection to the current graph are retained. Because of their presence in the original source network, cycles (e.g. feedback loops) can also be encountered in the generated topology.

As larger subnetworks are sampled from biological networks, the tendency to select similar subnetworks increases. In the extreme case that the subnetwork contains an equal amount of genes as the original network, the same (original) network would always be sampled. Therefore an additional *background network* is added to the network to increase the variation in the selected subnetworks. For a real biological microarray experiment it is generally assumed that only part of the genes on the chip are triggered by the conditions applied [174]. To take this fact into account, a *background network* is added that mimics pathways not elicited by the simulated experimental conditions. These background genes increase the dimension of the data set but are not themselves part of the network to be inferred. Their expression values are assumed to be

constitutive but change in a correlated way as a result of the biological noise modeled in the transition functions.

The topology of the combined network consisting of the foreground and the background network shows therefore more sampling variation than a single subnetwork of the same size would exhibit, while simulating some of the essential characteristics of real microarray data, namely a small set of genes that are triggered by the applied external conditions and a large set of genes that is constitutively expressed.

3.3.3 Characteristics of network topology generation methods

As will be shown in Section 3.8.2, the subnetwork selection method generates networks that more closely approximate the topological properties of biological networks than the tested random graphs do. Secondly, the cluster addition method generates networks that are closer to the source network than the networks generated with the neighbor addition method.

To evaluate the change in topological characteristics in function of the number of nodes in the subnetwork, networks of different sizes were selected using both methods (see Figure 3.4). The cluster addition method (Figure 3.4a) shows less variation for the median indegree compared to the neighbor addition method (Figure 3.4b). This is not surprising since adding a node and all of its children, as is done in the cluster addition method, better preserves the median indegree than adding a single node.

3.4 Transition functions

After generating the topology, transition functions representing the regulatory interactions between a transcription factor and the regulated genes are assigned to the edges in the network. In previous work, non-linear equations based on Michaelis-Menten and Hill kinetics have been used to model different types of local interactions between a gene and its parents [53, 74, 114].

Steady-state solutions of these equations are derived in order to define the transition functions between the genes. Biological noise is superposed on the kinetic equations, corresponding to stochastic variations in gene expression that are unrelated to the applied experimental procedures. By using the steady-state solutions, the generation of expression data scales linearly with the number of genes and therefore allows fast simulation of large networks comprising thousands of genes. However, by using a steady state solution rather than using the dynamic equations, the generation of dynamic expression data such as time series experiments, is excluded.

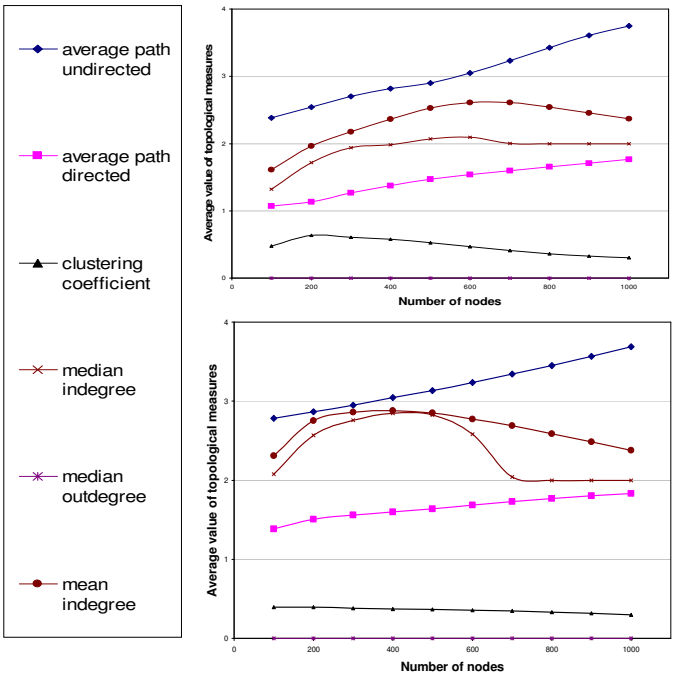


Figure 3.4: ***Topological characteristics of the resulting networks of the two subnetwork selection methods in function of the number of nodes.*** ***(a)*** *Cluster addition method;* ***(b)*** *Neighbor addition method.*

3.4.1 Interaction types

This Section provides a detailed description of the Michaelis-Menten and Hill kinetic equations that are used to model the regulatory interactions. An illustrative example is shown in Figure 3.5, illustrates the main concepts by means of one gene with a single activating regulator. The activator A binds to the promoter region P of its target gene. The transcription rate of the gene (v) is determined by the amount of activator that is bound to the promoter region, which is represented by the concentration (PA). The transcription results in a certain quantity of mRNA, represented as r. This mRNA also degrades at a certain rate k_d, which depends on the specific sequence of the mRNA transcripts, the concentration of specific digestion enzymes and several other factors.

Independent of the specific activation or inhibition pattern, the mRNA transcription rate is given by the following equation:

$$\frac{d[r]}{dt} = v - k_d[r] \tag{3.1}$$

where v is the mRNA production rate, $[r]$ is the concentration of the pro-

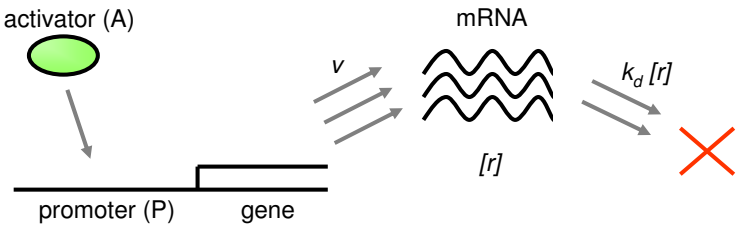
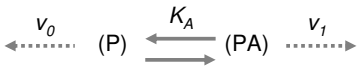


Figure 3.5: ***Illustration of an interaction between an activator and its target gene****. An activator A binds to the promoter region P of its target gene. The amount of activator that is bound to the promoter (concentration (PA)), determines the transcription rate v of the gene. The mRNA is digested by the cell at a certain rate* k_d*.*

duced mRNA and k_d is the degradation constant of the mRNA. In steady-state conditions, $\frac{d[r]}{dt}$ is 0 and therefore $[r] = \frac{v}{k_d}$.

In our example, the chemical equilibrium diagram for the binding of the activator to the promoter is given by (with dissociation constant K_A):



$$K_A = \frac{[P][A]}{[PA]} \tag{3.2}$$

The resulting mRNA production rate is given by Equation 3.3 where $[P]$ and $[PA]$ represent the concentration of unbound and bound promotor P with the activator A:

$$v = v_0.[P] + v_1.[PA] \tag{3.3}$$

Combining Equations 3.2 and 3.3 leads to the following expression for the mRNA production rate:

$$v = \frac{v_0^* + v_1^*.A/K_A}{1 + A/K_A} \tag{3.4}$$

where v_0^* and v_1^* denote v_0/K_A and v_1/K_A respectively. Concentrations such as $[A]$ and $[P]$ are replaced by their variable names A and P for notational convenience.

Various cases of transcriptional regulation exist and equations such as Equation 3.4 can be derived for (combinations of) each of these cases:

- *inhibitory*: the regulator is inhibitory (I) instead of activating (A).

- *cooperative*: a regulator is called cooperative when its binding affinity changes with the amount of regulator already bound (this is modeled with Hill equations).
- *competitive*: when regulators compete for the same binding place(s) on the promoter, they are called competitive.
- *synergistic*: two regulators are called synergistic when in case both regulators bind, the mRNA production rate is higher than the sum of the mRNA productions for the cases where only one of the regulators is bound.
- *antagonistic*: similar to the case of synergistic regulators, but the mRNA production rate is lower than expected in case both regulators are bound.

The mRNA production rates are given below for some examples of the above cases with two regulators:
One activator and one repressor, competitive:

$$v = \frac{v_0^* + v_1^* \cdot \frac{A}{K_a}}{1 + \frac{I}{K_i} + \frac{A}{K_a}} \tag{3.5}$$

One activator and one repressor, non-competitive:

$$v = \frac{v_0^* + v_1^* \cdot \frac{A}{K_a}}{(1 + \frac{I}{K_i}) \cdot (1 + \frac{A}{K_a})} \tag{3.6}$$

Two activators, synergism:

$$\begin{aligned} v_3^* &= \beta \cdot (v_1^* + v_2^*) \\ v &= \frac{v_0^* + \frac{v_1^* \cdot A_1}{K_{1a}} + \frac{v_2^* \cdot A_2}{K_{2a}} + \frac{v_3^* \cdot A_1 \cdot A_2}{K_{1a} \cdot K_{2a}}}{1 + \frac{A_1}{K_{1a}} + \frac{A_2}{K_{2a}} + \frac{A_1 \cdot A_2}{K_{1a} \cdot K_{2a}}} \end{aligned} \tag{3.7}$$

where β denotes the degree of synergism between the two activators.

A general equation can be derived for cooperative binding with N TFs (similar to [114]), covering all possible combinations of the above cases. Note that competitivity, synergism and antagonism are not modeled in this equation. The general steady-state equation for N regulators (P activators, Q inhibitors) is given by:

$$v = \frac{V_{0,max} + \sum_{i=1}^{P} (\frac{A_i}{K_i})^{n_i^{act}} \cdot \prod_{j=1}^{P \neq i} (1 + (\frac{A_j}{K_j})^{n_i^{act}}) \cdot V_{i,max}}{\prod_{i=1}^{P} (1 + (\frac{A_i}{K_i})^{n_i^{act}}) \cdot \prod_{j=1}^{Q} (1 + (\frac{I_j}{K_j})^{n_j^{inh}})} \tag{3.8}$$

The exponents n_i^{act} and n_j^{inh} are the Hill constants for each of the activators and inhibitors. Hill constants are integer and $n_i^{act} \geq 1$, $n_j^{inh} \geq 1, \forall i, j$.

In summary, for a transcriptional network with N genes, the following set of steady-state solutions for the mRNA concentration of each gene i is obtained:

$$[r^{(i)}] = \frac{v^{(i)}}{k_d^{(i)}} \tag{3.9}$$

where either the general Equation 3.8 is used for $v^{(i)}$ or one of the Equations such as 3.4, 3.5, 3.6 or 3.7 is used to describe a specialized interaction between a set of regulators and its target gene i.

Since the mRNA concentration for each gene $[r^{(i)}]$ depends only on the mRNA concentrations of its regulators, the determination of the individual concentration levels starts from the external inputs and percolates through the network from top to bottom. The computational cost for determining all the values therefore scales linearly in the number of genes in the network.

In the case of cyclic networks, an approximation is used where unknown cyclic inputs are initially given a random value and the above procedure is iterated where the concentration of each gene's mRNA level is determined given its regulators concentrations until convergence to a stationary regime. For cyclic networks, the computational cost is proportional to the number of genes in the network and to the number of iterations required for convergence. The number of iterations indirectly depends on the number of genes, the topology of the underlying network and the length of the cycles in the network. Typical values for the tested biological networks range between 100 and 1000. Note that oscillations can occur in this stationary regime and only one of the possible concentration values for each gene is sampled in those cases.

Choosing realistic parameter settings of these equations is a nontrivial task. Except for a few well characterized networks, no data about the parameters for the Michaelis-Menten and Hill functions is available. Therefore, the value of each parameter is chosen from a distribution that allows a large variation of interaction kinetics likely to occur in true networks (including linear activation functions, sigmoid functions, ...), while avoiding very steep transition functions. An example of the possible interactions is given in Figure 3.6 for a gene which is regulated by a single activator. In realistic networks, often more complex situations occur with multiple regulators that exhibit inhibitory and/or activating regulation and possibly competitive or co-operative behavior.

3.5 Sampling data

In this section we describe how a gene expression data set is obtained by simulating the synthetic network under different simulated experimental conditions.

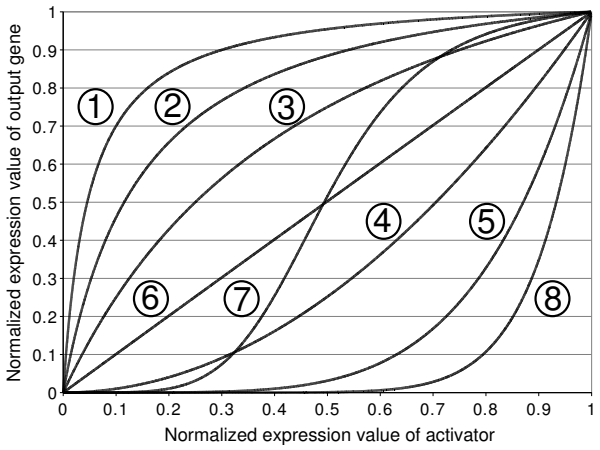


Figure 3.6: ***Interaction functions for one activator****. Examples of interaction functions that describe the influence of one activator on the expression level of the regulated gene for different combinations of the kinetic parameters K and n_{Hill}. Parameter tuples (K,n_{Hill}) are (0.05,1); (0.15,1); (0.5,1); (10,2); (10,5); (100,1); (0.5,5); (100,10) for functions 1 to 8 in ascending order.*

3.5.1 Generating gene expression data

When generating data, we assume that the expression of the genes depends on how changes in external conditions trigger the network. External conditions are modeled by choosing a gene set without regulatory inputs and setting their expression level to a value different for each experiment, in a simulated response to changing experimental conditions. Remaining genes without regulatory inputs are assigned a random constitutive expression level.

The expression levels of the genes in the network are subsequently calculated, as specified by their transition functions, starting from the input genes. For genes involved in cycles, it is possible that not all inputs of their transition function are known during propagation of the values through the network. To model these loops, an approximation compatible with the steady-state transition functions was chosen: each edge in a cycle is modeled as a regular steady-state interaction. In each simulated experiment, genes that have an undefined input are initially assigned an arbitrary expression value and calculations for the entire network are repeated until transient effects have disappeared before generating the output expression values. In case of oscillatory behavior, the expression data is taken from an arbitrary point in the period, mimicking the situation in a real microarray experiment.

3.5.2 Adding noise

After sampling from the network, a data set with mRNA expression levels for all genes is obtained for different simulated conditions. All gene expression values are normalized between 0 and 1, where 0 indicates that no transcription occurred and 1 refers to a maximal level of transcription. Besides the biological noise, microarrays are subject to random experimental noise. This experimental noise is added to the simulated microarray data and is approximated by a log-normal distribution [138]. Consistent sources of variation (dye effects, array effects, ...) are not explicitly modeled as they can be removed in real data by an adequate preprocessing step.

3.5.3 Simulated expression data

Figure 3.7 gives a representative example of a network topology of 50 genes obtained by selection from the *E. coli* source network [108] using the cluster addition method. From this network, gene expression data was generated for 100 simulated microarray experiments.

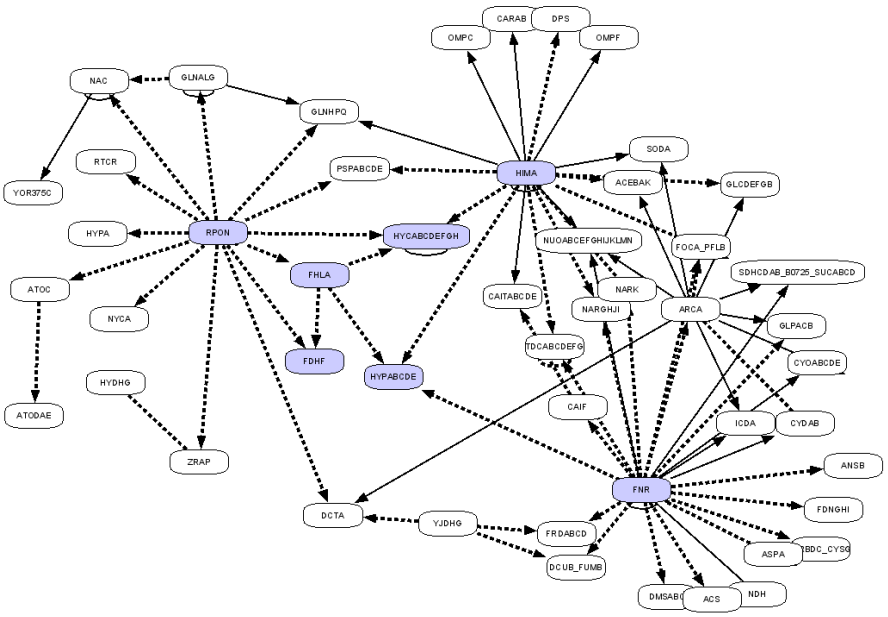


Figure 3.7: ***Randomly chosen example network with 50 nodes****. The cluster addition method was used for subnetwork selection from the E. coli network. Dashed edges indicate activation, full edges indicate repression.*

Three input genes were defined ($g1$,$g2$, $g3$) for which setting the expression

values corresponds to changing external conditions in a microarray experiment. Distinct external conditions were thus mimicked by randomly choosing the expression values between 0 and 1 for each experiment.

In Figure 3.8, part of the simulation results of the example network are shown. The selected part of the example network is shown in Figure 3.8a and is also indicated by the shaded nodes in Figure 3.7. Figures 3.8b-d show how each of the input genes affects its direct children. For example, *g1* has a strong effect on both *g4* and *g5*, but a less pronounced effect on the expression level of *g7* since *g7* has a repressor feedback loop and is also stimulated by another input gene *g2*. In Figure 3.8d, the expression levels of *g2* are also added to illustrate the relation between two independent genes like *g3* and *g2*.

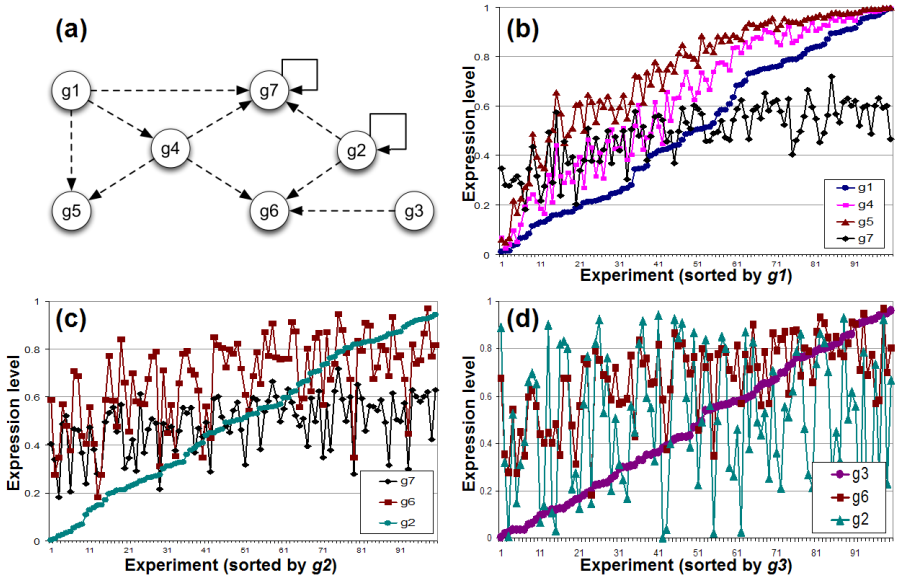


Figure 3.8: ***Simulation results for a subset of genes of the example network (labeled g1 to g7).*** *This subnetwork has three input genes and contains a repressor feedback loop for* g7*. The X-axes shows the different conducted experiments, which are sorted according to the expression value of each of the input genes. The Y-axes shows the normalized expression values for the genes directly regulated by the input genes, except for (d) where the expression value for* g2 *is also shown.*

3.5.4 Generator parameters

To benchmark an algorithm, having access to data sets of an increasing level of difficulty is useful. Experience shows that in real data, the difficulty of the structure learning task of an inference problem is influenced to a large extent by the topology of the network to be inferred and by the type of the regulatory

interactions present. For example, more data is required to resolve interactions that are not fully exercised [151].

Initial performance testing of an algorithm can be done on rather easy problems (e.g. small, noiseless networks without synergism or cooperativity between regulators). Increasingly difficult data sets can then be generated to further optimize the inference method.

The following parameters controlling the gene network generation and sampling process are user-definable:

- The size of the network in number of nodes.
- The number of background nodes.
- The number of available experiments and samples for each condition.
- The level of stochastic and experimental noise.
- The fraction of complex interactions.

3.6 Inference algorithms: ARACNE, SAMBA, Genomica

In Section 3.8, three different network inference algorithms are characterized using simulated datasets generated with SynTReN. A brief overview of these methods is given below.

Genomica

Genomica [147] uses expression data to construct a network of interacting modules, each consisting of co-regulated genes, their regulators, and the conditions under which regulation takes place (i.e. the regulatory program). The method is based on probabilistic relational models [94, 146], a relational extension to Bayesian networks.

SAMBA

SAMBA [154] is a bi-clustering algorithm that groups genes by means of a clustering of similar expression patterns in the input data over a subset of input conditions. The algorithm is based on a graph theoretic approach and statistical modeling of the data. The subsets of genes that jointly respond to specific conditions can be interpreted to form a module network.

ARACNE

ARACNE [14] explicitly infers a gene interaction network from microarray expression profiles on the basis of *mutual information* [124] between the genes. Two parameters control the pruning of candidate interactions. The *mutual information (MI) threshold* eliminates edges that have low mutual information and thus removes edges between genes that have a nonzero MI solely due to random sampling of the data samples. The *data processing inequality (DPI)* is used to eliminate the least strong interaction of a triplet of interactions and thus for example eliminates transitive interactions between two genes if the interaction is indirectly through a third gene.

All inference algorithms were run at their default parameter settings. The only exception to this rule was ARACNE, which requires the specification of the DPI threshold parameter. The authors suggest a DPI threshold level between 0 and 0.15. In our experimental setup, a DPI level of 0.10 was used as a typical threshold and a DPI level of 0 was used for comparison (all indirect interactions are retained).

3.7 Performance evaluation criteria

In this section we discuss the score metrics that were used to indicate the quality of the inferred gene interaction networks. The optimal solution to the inference problem is a reconstruction of the complete interaction network with its topology and causal relations between genes. In reality exact reconstruction of the causal relations in the network is impossible, since generators like SynTReN produce steady state data. To infer causality additional information is required such as time-course gene expression data, perturbation experiments with gene knockouts or other types of data sources such as location data and motif data. As a consequence, the input network topology has to be transformed to an undirected network to allow comparison with the reconstructed networks. Additionally, both Genomica and SAMBA generate module networks, which are a partitioning of the genes in subsets that belong together. This modular output has to be translated into a corresponding gene regulatory network to allow comparison with the original interaction network, which does not explicitly contain the concept of modules. To this end both the known network topology and all of the reconstructed networks are converted into a gene-by-gene binary adjacency matrix for further analysis.

For the original SynTReN network, this matrix is constructed by calculating the shortest undirected path length between every pair of genes in the original network, and comparing this distance to a threshold. If the distance is below the threshold the corresponding two entries (due to symmetry) in the adjacency matrix are set to 1. If a threshold '1' is used this procedure results in a matrix representing an undirected version of the original network. However, because

of the way the adjacency matrix is constructed for the module networks, it is necessary to use a higher threshold to allow meaningful comparison. By default a threshold '2' was used, in effect grouping genes that are linked by at most one intermediary gene into a 'module'.

For the module networks, the adjacency matrix is constructed as follows: starting from the gene-by-gene identity matrix M, the entries M_{ij} and M_{ji} are assigned a value of 1 if at least one module is present that contains both genes i and j. A regulatory module therefore corresponds to a clique in the graph, with a connection between every pair of genes in the module. In this way, the modules will probably contain many indirect interactions. If genes a and b share a common regulator c and all three genes are members of the module X, the adjacency matrix will contain non-zero entries for the indirect interaction between a and b. Apart from a set of modules, Genomica also generates a regulatory program for each of the modules. While this is information is certainly valuable, it was not used in the presented analyses primarily to preserve a common evaluation ground between the different algorithms.

Because ARACNE directly infers an interaction network, rather than a module network, the above procedure to construct the adjacency matrix for the original network actually rewards the algorithm for finding indirect interactions. However, ARACNE is capable of efficiently pruning these indirect interactions based on the MI between each pair of genes. By lowering the path length threshold to a value of 1 when constructing the adjacency matrix from the original network, it is possible to evaluate this ability, and effectively require ARACNE to infer the exact – but undirected – original network topology. ARACNE's output can be transformed into an adjacency matrix in a straightforward way because it already contains a single p-value for each selected pair of genes that share a regulatory interaction in the inferred network. Genes that are not connected according to ARACNE do not appear in its output. Therefore, the selected gene pairs correspond exactly to the 1's in the adjacency matrix, while all other entries are 0, except for those on the diagonal, as mentioned before.

Comparison between the resulting adjacency matrices was performed by counting the corresponding and conflicting entries in both matrices and calculating *sensitivity* (also known as *recall*), *specificity*, and *precision* (also known as *positive predictive value*). As a summary metric, the *F-measure* was used, which is the harmonic mean of precision and recall. We refer to Appendix A.2 for a more detailed analysis of the performance metrics.

3.8 Results

3.8.1 Experimental setup

The main goal of the presented analyses is to demonstrate how the use of synthetic data can provide substantial insight into the performance of a network inference algorithm and its relationship to properties of the input data. The experimental setup that will be used, is shown in Figure 3.9: in a first step, a synthetic gene interaction network is generated based on a chosen network topology and interaction type. Next, expression data is generated that corresponds to the gene interactions dictated by the network. This involves setting various levels of noise and structuring the resulting dataset in experiments and samples per experiment. An *experiment* in this context involves the selection of a set of external conditions which are subsequently perturbed and fed into the transcriptional gene network. For each experiment, SynTReN produces a number of microarray datasets, referred to as *samples*. The resulting dataset is then used as input to a number of different network inference algorithms, which produce a candidate network of genes or gene modules. In a final step, both the original network topology and the inferred candidate are compared through a derived adjacency matrix, and the calculated performance metrics are summarized by means of plots and discussed in the following sections.

The resulting performance metrics are used only as a relative score to differentiate across different experimental settings. For several reasons, they were not intended to quantitatively assess the performance of the inference algorithms or to compare algorithms with each other. First of all only default settings were used when running the inference algorithms. These settings are most likely not the most optimal parameter settings for every experiment. Second, the metrics only assess the presence and absence of edges in the inferred network. A more sophisticated or in-depth evaluation of the inferred networks could give more insight in the performance of an algorithm for a specific experiment, but it less suitable for the high-throughput nature of this study.

With the setup described above, we aim at investigating the following questions related to several parameters of the expression data that will be supplied to the inference algorithms:

Network size: how trustworthy are inference results of ever larger interaction networks, given abundant expression data? This is important since it relates to the applicability of inference methods for large networks, and links to the discussion of the need of heterogeneous data sources to infer truly large networks.

Graph topology: to what extent does the quality of inferred networks depend on the statistical nature of the topology of interaction networks? Several previous studies [112, 114] have reported the use of synthetic expression data from random graph models to validate network inference. However, previous work

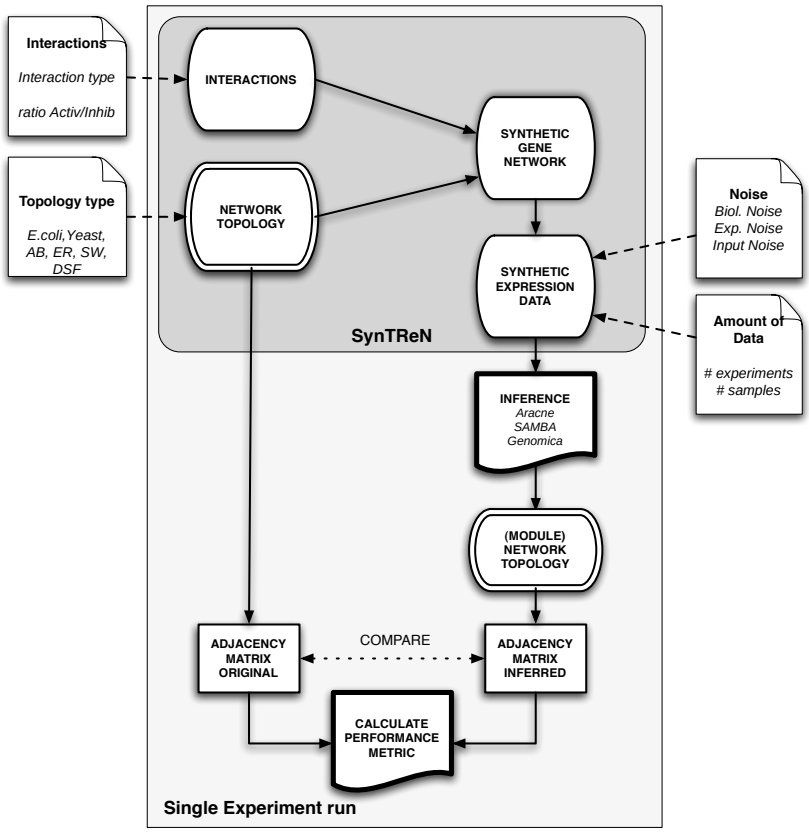


Figure 3.9: ***Overview of the experimental setup used****. A synthetic interaction network is generated and expression is data derived, which is used as input to several inference algorithms. The inferred networks are then compared to the original by means of calculations on corresponding adjacency matrices. Parameters to be defined are shown on the outer edge of the diagram.*

has shown that these random graph models do not resemble real biological networks in every respect [160]. This raises the question of the performance of algorithms that have been designed with only random network topologies as a benchmark. To what extent does good performance on a specific class of networks generalize to other classes? Can increasing knowledge of topological properties interaction networks substantially increase the ability to design better inference algorithms?

Noise type and amount: what is the effect of various types and amounts of noise in expression data on the quality of inference results? The experiments give an indication to the added value of a reduction of noise in high-throughput experiments. Other experiments try to answer the relation between the esti-

mate of biological noise in datasets and the confidence expressed in inference results.

Amount of expression data: what is the marginal gain in inference quality by spending resources on obtaining extra datasets of microarray experiments? Do inference algorithms reach a maximum inference quality, after which supplying more expression data becomes pointless? The reported experiments investigate this issue in relation to the amount of noise present in the expression data provided.

Interaction type: what is the impact of different interaction types between genes? More specifically: to what extent do highly non-linear interactions act as a buffer to mask the activity of downstream genes in an interaction cascade? Should such a buffering effect occur, it can be expected that inference results downstream of such genes are of lower quality, which can be taken into account when validating results on real-world data.

3.8.2 Validation of biological subnetwork selection methods

To validate our approach, a series of synthetic networks is generated both by using different types of random graph models (Erdös-Rényi (ER), Watts-Strogatz or small-world (SW), Albert-Barabási (AB) and directed scale free (DSF) random graph models) and by selecting subgraphs according to the methods described in Section *Network topology*. To obtain representative sets of networks for the given models, a sweep was done across a large range of possible parameter settings for the tested models. The topological properties of each of these networks are compared to these of the complete *E. coli* and *S. cerevisiae* networks.

In Figure 3.10 and 3.11, the average indegree is plotted versus the average directed path length to illustrate the different characteristics of the random graph models, the previously described TRNs and the selected subnetworks.

Figure 3.10 shows that it is not possible to choose the parameters for Erdös-Rényi and Albert-Barabási random graph models such that both the average directed path length and average indegree are simultaneously close to the values of biological TRNs, although they can resemble biological networks for a single topological measure. Similar results are obtained for evaluated topological characteristics other than average indegree and average directed path length, such as average clustering coefficient, average out-degree and average undirected path length (results not shown).

From Figure 3.11, a similar conclusion can be drawn for SW networks. They can resemble biological networks for a single topological measure, but not for several measures simultaneously. DSF graphs [24] can however resemble biological networks with respect to both average directed path length and average indegree for well-chosen parameter settings. Again, similar results are obtained for other topological characteristics (results not shown).

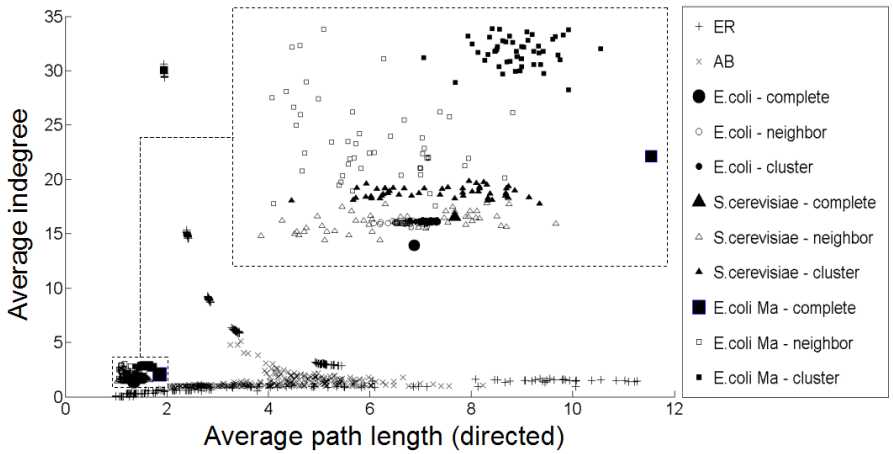


Figure 3.10: ***Topological properties of ER and AB random graphs, for subselected networks of 300 nodes and for the complete biological networks****. Average indegree versus average directed path length for ER and AB random graphs of 300 nodes and for biological networks. Biological networks are the complete E. coli (both networks described [149] and [108]) and S. cerevisiae network, indicated by the suffix '-complete'. Subnetworks containing 300 nodes were created by both the neighbor- and cluster-addition method, indicated by the suffixes '-neighbor' and '-cluster' respectively. The region of the biological networks is enlarged in the upper right corner of the figure. ER and AB random graphs exhibit a phase transition [23, 44]. For low connectivity, often no path exists between several pairs of nodes and many path lengths are therefore infinity. These are not considered for calculating the average path length, which therefore appears small because it is calculated from the few short paths that are present. When increasing the p-value (the probability of having an edge), the paths are increasingly made up of more edges until a point is reached where the graph starts forming one giant network. Adding more edges then increases the density of the graph connections, resulting in a decrease of the average path length. ER: Erdös-Rényi, AB: Albert-Barabási.*

3.8.3 The effect of network size

To assess the influence of network size on inference performance, subnetworks of different sizes were selected from the *E. coli* source network [149], according to the cluster selection method. The size of the selected subnetworks varied from 50 to 300 edges in intervals of size 10 and linear interaction functions were assigned to all edges in the network. All topnodes acted as external input genes. A large expression dataset with a low noise level (1000 experimental conditions, 1 sample each, 0.05 experimental noise) was generated for each of the networks. For each algorithm the F-measure was plotted in function of increasing network size. All three algorithms show clearly different quantitative behavior when faced with larger networks.

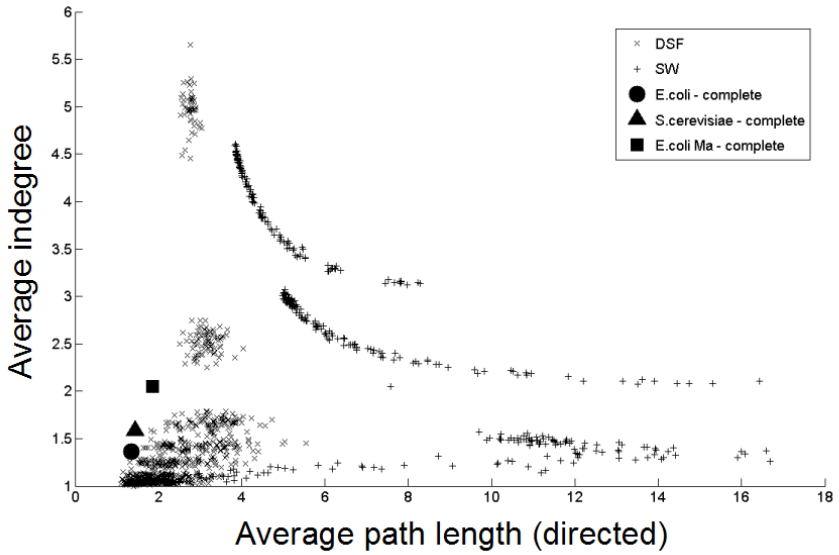


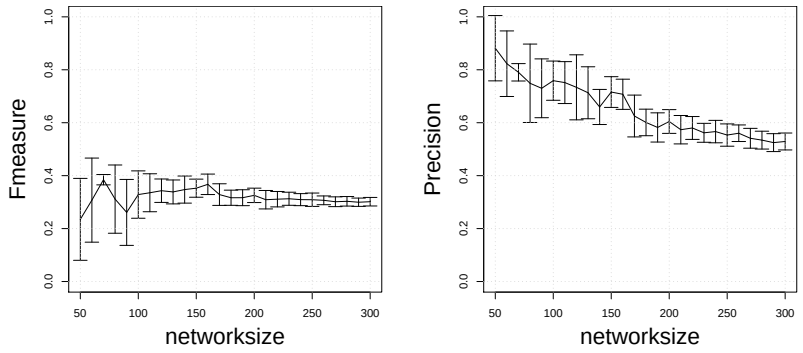
Figure 3.11: ***Topological properties of DSF and SW random graphs of 300 nodes and the complete biological networks****. Average indegree versus average directed path length for DSF and SW graphs of 300 nodes and biological networks. Biological networks are the complete E. coli (both networks described by [149] and [108]) and S. cerevisiae network [65]. SW: Small world (Watts-Strogatz [166]), DSF: Directed scale free (Bollobás [24]).*

As illustrated in Figure 3.12a, SAMBA's overall performance as indicated by the F-measure, stays the same regardless of size, although there is clearly a larger variation for smaller networks. Its precision, however, decreases with increasing network size (see Figure 3.12b). Genomica produces lower F-measure scores as the networks to infer grow, but levels out eventually (Figure 3.12c). Genomica did not output results for many of the larger networks at the settings that were used, even after relatively long running times. In cases where the running time exceeded 12 hours, the experiments were stopped and these data points are missing from the analysis. ARACNE's performance also decreases for larger networks, but does not seem to level out within the tested range (Figure 3.12d).

For each of the three algorithms, we observe that the variance on the F-measure decreases with increasing network size (Figure 3.12). The reason is that for larger subnetworks from the *E. coli* network, more often similar parts are selected in the subnetwork selection methods. The cluster addition method which was used in these experiments is more sensitive to this effect than the neighbor addition method since it adds a complete hub of genes (a gene and all its targets) rather than a single gene during the network generation. For

large subnetworks this inevitably leads to the selection of the same large hubs for the different networks.

In summary, inference performance drops as network size increases, even if enormous amounts of data are available. This decay however is not as drastic as what might be expected, especially for ARACNE which shows no sign of substantially reaching the end of its application scope for larger networks. In order to infer larger networks with high confidence, it seems that complementary approaches such as adding additional data sources or incorporating domain knowledge are needed.



(a) F-measure - SAMBA (b) Precision - SAMBA

(c) F-measure - Genomica (d) F-measure - ARACNE

Figure 3.12: ***Impact of network size on the performance (F-measure and precision) of ARACNE, Genomica and SAMBA.***

3.8.4 The effect of network topology

In the following series of experiments smaller networks of approximately 50 genes were used. The effect of graph topology on the performance of the inference algorithms was studied by creating gene networks with a topology

derived from different random graph models that are known to approximate biological networks. The studies models are the Erdös-Rényi [52] (ER) random graph model, the Albert-Barabási [5] (AB) scale-free network model, the Watts-Strogatz [166] (WS) small-world model and the directed scale free (DSF) model as described by Bollobás [24]. Apart from these random models, a set of topologies was also generated by selecting subnetworks from the previously described *E. coli* [149] and *S. cerevisiae* [65] transcriptional network, as described in [160]. These latter graphs more closely approximate the topological properties of known transcriptional networks [160].

For each random graph model a small number (approximately 10) of representative graphs topologies was created by sweeping the model parameters across a range of values. The parameter values were varied around a default set of values that was chosen to produce graphs whose properties are close to the *E. coli* network (this was investigated in a previous study [160]). For every generated graph topology, a series of 10 experimental runs was performed. In each run several distinct synthetic gene networks with the same network topology were created by assigning linear interaction functions to the edges in the graph (see also Section 3.2) and synthetic gene expression data was generated for each of the resulting gene networks. Each dataset consisted of 100 simulated experimental conditions, with bionoise set at 0.05, and no experimental or inputnoise. For every inference algorithm about 5600 expression datasets were supplied for inference, covering 92 different network topologies.

Figure 3.13a indicates that SAMBA's performance does not differ markedly between any of the graph classes with regard to sensitivity or specificity. Sensitivity is low across graph classes, because few true positive interactions are found.

Both Genomica and ARACNE (Figures 3.13b-c) achieve very similar sensitivity and specificity values for both AB and WS graphs, causing these graph models to be largely co-located on both the plots (upper left). Other random graphs, like the ER and DSF graphs, cover a much wider range of sensitivity/specificity combinations. The *S. cerevisiae* and *E. coli* subnetworks also show significant overlap with each other but clearly cover a different sensitivity/specificity than the AB and WS graphs.

For both Genomica and ARACNE it is interesting to note that some of the *E. coli* and *S. cerevisiae* subnetworks show a specificity of zero at high sensitivity. This is due to the fact that the particular network topologies involved – which were in all cases subnetworks selected with the cluster addition method – are in fact one big star-like structure, with a single regulator regulating a large number of genes. Since all gene-pairs are separated by at most two edges, the adjacency matrix for a path length threshold of two is an all-one matrix. As a consequence, the number of true negative interactions will always be zero and thus result in a specificity value of zero.

Summarized, the topology of the network can have a strong impact on the

performance of an inference algorithm. This should be taken into account when evaluating inference algorithms using synthetic datasets. It is encouraging to note that for two of the algorithms tested here, namely Genomica and ARACNE, the inference results on topologies that are known to be biologically more plausible, are better.

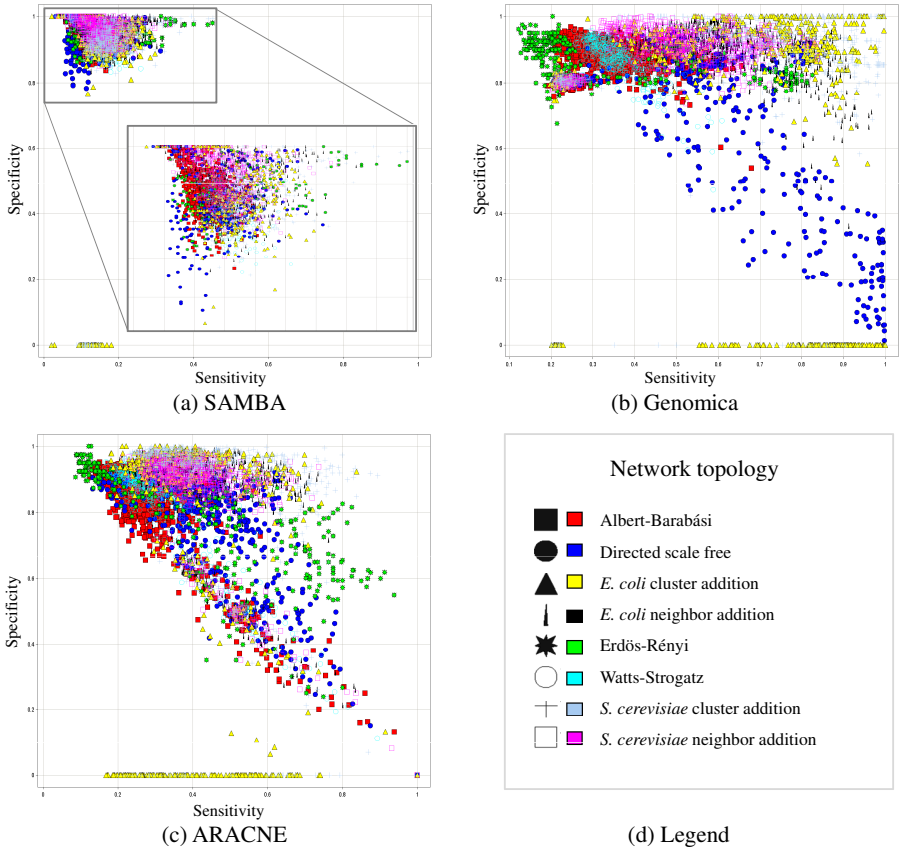


Figure 3.13: ***Impact of network topology on sensitivity and specificity for ARACNE, Genomica and SAMBA.***

3.8.5 The effect of various noise types

The impact of noise was investigated by generating expression data from the same network under a variety of noise conditions. The network topology that was used to generate data, was a subnetwork selected from *E. coli* [149] with 50 genes and 76 edges in which 5 genes acted as external inputs. In each of the 20 experimental runs, linear interaction functions were assigned to the edges in the network and expression data was sampled for a range of noise levels.

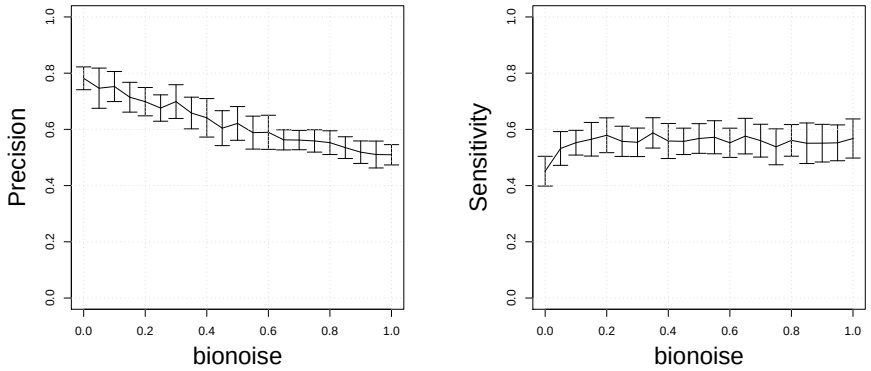
The tested range varied from 0.0 to 1.0 in intervals of 0.05 for each of the noise types (bionoise, experimental noise and inputnoise). The nominal dataset consisted of 100 experimental conditions, represented by 200 array datasets (2 samples per condition). In the next paragraphs, not all figures are shown. As a guideline for the discussion, note that for a linear interaction sampled at a 1.0 noise level, the correlation between the regulator and the regulated gene is still quite high (approximately 0.7), if the expression data sufficiently spans the range of possible regulator values. This means that at a noise level of 1.0, the data is still quite correlated compared to genuinely random data.

Since the effect of inputnoise only manifests itself as a difference between multiple samples measured under the same external conditions, its effect is very small under the given experimental conditions for each of the three algorithms. A smaller number of experimental conditions and a larger number of samples can possibly provides more insight, but was not adopted in the experiments to limit the computational costs.

In the results for SAMBA, the F-measure linearly decreases with increasing levels of bionoise. A similar decrease in F-measure is observed with increasing experimental noise. Detailed analysis shows a gradual increase in precision to a level of 1 – the level at which all inferred interactions are correct – reflecting the fact that SAMBA outputs module networks with very small or even empty modules at high noise levels. In that case the only non-zero entries left in the adjacency matrix are those on the diagonal which results in a high precision. The sensitivity however drops to zero at high noise, globally resulting in decreasing F-measure as discussed. The effect of inputnoise on the performance is much less pronounced as discussed.

In Genomica's case, the F-measure shows a very small decrease across the tested range of bionoise and experimental noise levels. Precision (Figure 3.14a) decreases linearly with increasing bio- or experimental noise, while sensitivity remains at an almost constant level (Figure 3.14b). The highest F-measure values are not achieved with zero noise, but with a low amount of noise. A possible explanation for this observation is the fact that Genomica uses a correlation based clustering as an initial step to infer the module network. In zero noise conditions such a correlation based approach has the tendency to overconnect the network, because indirectly interacting genes will show as strong a correlation as those that are directly interacting. Similar behavior was observed when testing other correlation based methods, the data of which is not included in this study.

For ARACNE, the effect of noise on the quality of the inferred output network depends strongly on the scoring procedure. When using the default procedure (see Section 3.7), the F-measure shows a reverse-S-shaped decrease in F-measure with increasing levels of bionoise (Figure 3.15a). This decrease is much more linear for experimental noise (Figure 3.15c). For ARACNE as well, inputnoise has a less pronounced effect (value around 0.7) although the



(a) Impact of bionoise on the precision for Genomica.

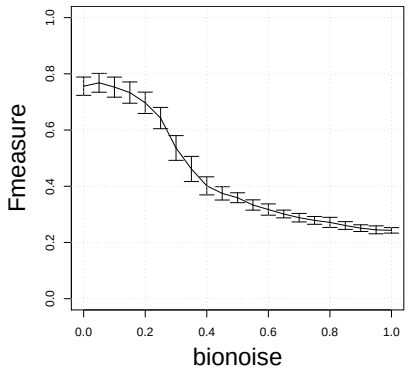
(b) Impact of bionoise on the sensitivity for Genomica.

Figure 3.14: ***Impact of bionoise on performance metrics of Genomica.***

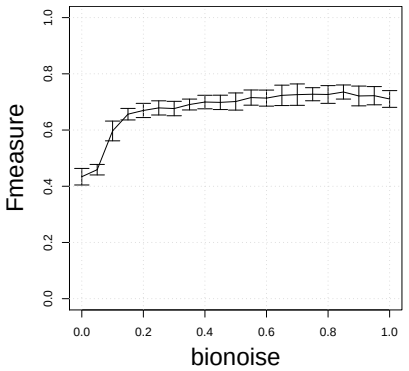
F-measure scores show a very small decrease at high noise levels. On the other hand, as explained in Section 3.7, it is possible to require ARACNE to precisely infer the original network and evaluate its ability to remove indirect interactions. In this set-up, the algorithm's DPI-threshold was set to 0.00 for most strict pruning of indirect interactions. In this case the F-measure (Figure 3.15b) shows a very different behavior compared to the settings used above. The performance metrics, which are at a low level when little or no bionoise is present, show a steep *increase* with rising levels of noise to a plateau from which they then decline very slowly. A possible explanation for this behavior is that when almost no noise is present the mutual information (MI) between indirectly connected genes is almost equal to that of the direct interactions, while the propagating bionoise will tend to decrease the MI between indirectly connected genes relative to those that are directly connected, resulting in more efficient pruning. In this setting the effect of experimental and inputnoise is also quite different: the F-measure remains unchanged and at a low level (see Figure 3.15d), with an increase in precision offset by a decrease in sensitivity. Possibly, ARACNE can not prune the indirect interactions efficiently in this case, because these types of noise do not propagate through the network and tend to decrease both the MI of the direct and that of the indirect interactions in equal amounts.

ARACNE is quite robust in the face of increasing bionoise. When no bionoise is present, the performance is high if the algorithm is not penalized for indirect interactions (Figure 3.15a). With much bionoise, the performance remains high – and in a way becoming even better – because indirect interactions are effectively removed. For experimental noise on the other hand, performance degrades gradually under the first scoring regime (Figure 3.15c). However, this is not due to the removal of indirect interactions as with bionoise, since

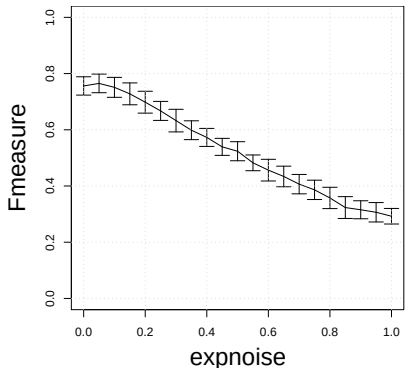
performance is low across the entire noise range for the second scoring regime (Figure 3.15d).



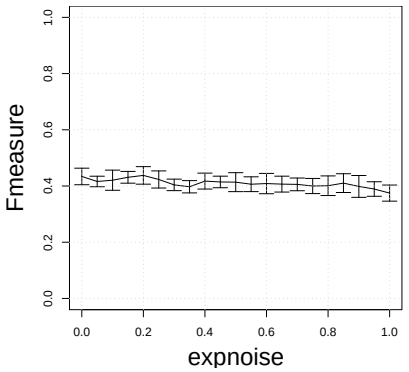
(a) Impact of bionoise, default scoring regime.



(b) Impact of bionoise, second scoring regime.



(c) Impact of experimental noise, default scoring regime.



(d) Impact of experimental noise, second scoring regime.

Figure 3.15: ***Impact of different noise types on performance measures for ARACNE*** *A definition of the scoring regimes is given in Section A.2.*

In summary, a clear effect of bionoise and experimental noise on the inference performance of all algorithms is observed, with inputnoise having the smallest influence. This illustrates that noise is an important, and sometimes required factor for network inference. All three noise types available in the SynTReN generator provoked a qualitatively different inference behavior, which supports their adoption in generators for synthetic data in general.

3.8.6 The effect of available expression data

This series of experiments aims at showing the impact of available expression data on inference results. The *E. coli* subnetwork with cluster addition of size 50 genes was chosen as the source network topology and 20 experimental runs were done by assigning linear interaction functions to the edges and subsequently sampling data from the network. Datasets varied in size form 5 simulated experimental conditions to 200 with a step size of 5. To evaluate the dependency between the number of arrays that are needed to infer the network, and the amount of noise in the dataset, the simulated expression data contains ranging experimental noise levels from 0 to 1 with step size 0.2. All other parameters were kept constant during the experiments.

A selection of Figures is shown to guide the discussion. Figures 3.16a-d show the F-measures for SAMBA, Genomica and ARACNE, and the precision for Genomica respectively. Error bars were calculated but are not shown on the plots to improve visibility. The relative size of the error bars was comparable to those of the preceding Figures such as Figure 3.15.

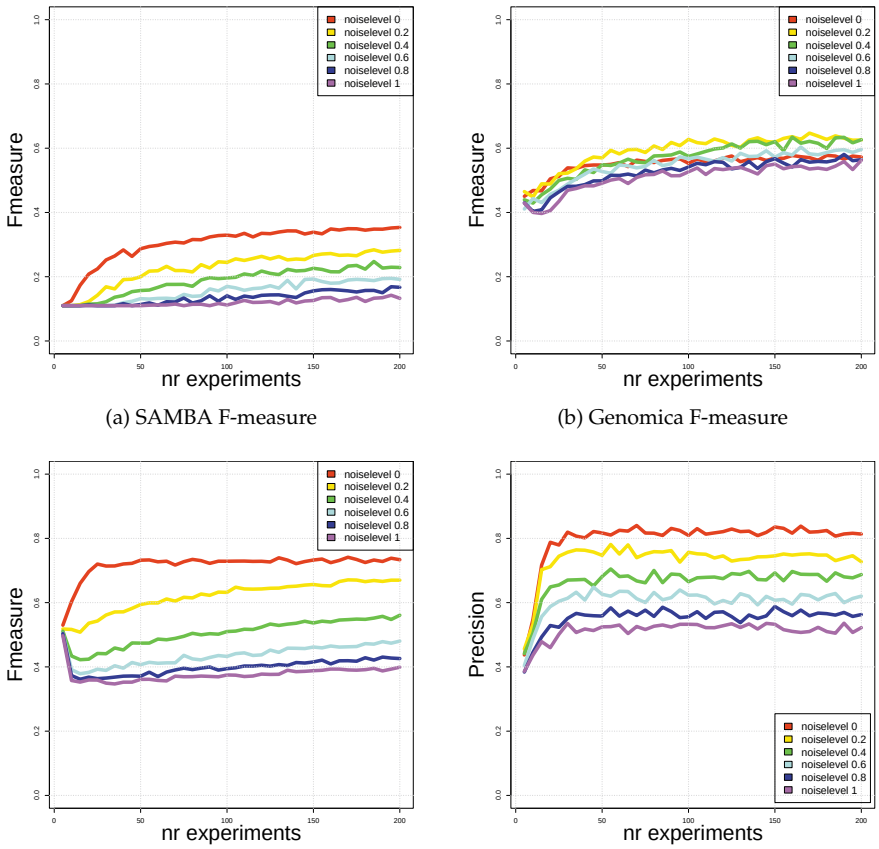
The behavior of the tested algorithms to increasing data and noise differs substantially, so we start by splitting up the discussion per algorithm. As shown in Figure 3.16a, SAMBA's F-measure scores increase gradually when the algorithm is presented with increasing amounts of expression data to infer the network. No performance plateau was reached within 200 experiments, and this behavior is consistent across noise levels. A more detailed analysis (plots not included) shows that precision gets lower with an increasing amount of data, but is offset by an increase in sensitivity, leading to the increasing F-measure on Figure 3.16a. This implies that more interactions are found, of which an ever increasing ratio are false (precision drops) but this is countered by an increasing ratio of correctly inferred interactions (sensitivity increases) leading to the overall better performance of the F-measure. The higher the noise level, the consistently lower the inference quality.

For Genomica however (Figure 3.16b), a similar increase in F-measure is noted for increasing input data, but a plateau a reached after which further addition of expression data does not result in significant changes in performance. With increases in experimental noise this behavior is generally maintained, only the maximum score achieved is slightly lower. An interesting effect occurs however when examining the score for zero noise: as shown in Figure 3.16b the highest F-measure values are reached for a noise level of 0.2, not in absence of noise as might be expected. This is due to a significantly lower sensitivity at a noise level of 0 than at small positive noise (plot not shown). Precision values on the other hand are ranked according to noise level as expected, with the highest values for 0 noise, as shown in Figure 3.16d.

The results for ARACNE in Figure 3.16c show that it performs quite well even with relatively small datasets, reaching maximum plateau performance

quickly. When noise is introduced however, performance behaves qualitatively different, requiring larger amounts of data to climb across the entire tested range without reaching a plateau.

The conclusion for this experiment is that there is a substantial but decreasing benefit of supplying more expression datasets when trying to infer interaction networks. The algorithms tested behave differently, sometimes reaching a maximum performance such as ARACNE in noiseless datasets or Genomica although the achieved score was lower than ARACNE's. Taking the special case of ARACNE inferring from noiseless data apart, reaching the inference plateau takes enormous amounts of data, given the fact that the target network aimed for here consists of only 50 genes.



(a) SAMBA F-measure

(b) Genomica F-measure

(c) ARACNE F-measure

(d) Genomica precision

Figure 3.16: ***Impact of available expression data on performance measures for ARACNE, Genomica and SAMBA.***

3.8.7 The effect of different interaction types

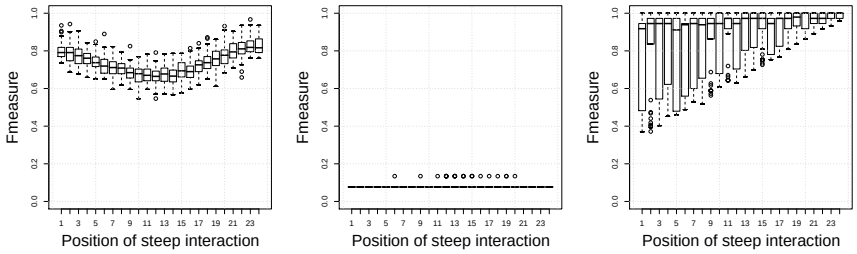
This section describes an experiment to assess the impact of different interaction types between genes. The extent to which a highly non-linear interaction can act as a buffer to mask the activity of downstream genes in an interaction cascade was examined. Should such a buffering effect occur, it can be expected that inference results downstream of such genes are of lower quality, which can be taken into account when validating results on real-world data. In order to observe this effect, a specific experimental setup was designed: chain topologies of varying length (8, 15 and 25 genes) were created, with only the first node of the chain acting as an external input. Only the results for chain-length 25 are shown. Two types of experiments were performed:

1. In the first experiment linear interactions were assigned to all edges in the network except for one, which was assigned a steep interaction. A steep interaction acts as a kind of threshold, with regulated nodes having a nearly constant expression value for most of the expression range of the regulator, and showing a large change in expression value for a small range of input values. It was hypothesized that this would have a negative effect on the ability to infer the presence of this interaction, because the correlated changes will only be observed in a relatively small number of sampled experiments.

2. In the second type of experiment, an increasing number of edges in the chain were assigned steep interaction functions, starting with the first edge in the chain and continuing down the chain for each additional steep interaction. For every combination that was tested, 50 runs were performed with a biological noise level of 0.05.

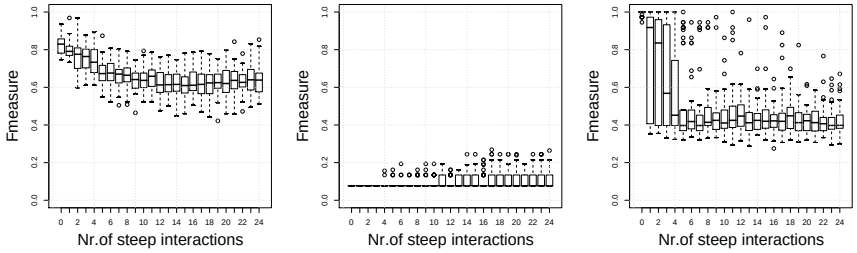
The particular experimental setup requires the performance measures of the different algorithms to be adapted. For SAMBA and Genomica, that aim at inferring modules and module networks, the scoring mechanism was altered such that the algorithms were required to assign all the genes to one large module to obtain a high score. In practice, this means that the adjacency matrix related to the original network was set to a matrix containing only non-zero entries. ARACNE, on the other hand, was scored on its ability to precisely infer the chain topology.

The F-measure for SAMBA is low overall (around 0.1), independent of the number or position of steep interactions (see Figures 3.17b and 3.18b). For the experiment with one steep interaction, Genomica achieves the highest F-measure scores when the steep interaction is at the beginning or at the end of the chain (see Figure 3.17a). A possible explanation is that a steep interaction in the middle of the chain will tend to lower the correlation between the genes upstream of the interaction and the downstream genes. In this case, Genomica often breaks up the network in more than one module, with the



(a) Genomica (b) SAMBA (c) ARACNE

Figure 3.17: ***Impact of a single steep interaction in a linear chain on the performance of ARACNE, SAMBA and Genomica****. Impact on the performance (F-measure) of ARACNE, SAMBA and Genomica of one single steep interaction at different positions in a linear chain of 25 single-input single-output genes with linear interactions. The plot shows a series of boxplots for each position in the chain where a linear interaction was replaced by a steep interaction. Each of the boxplots is the summary of 50 independent runs with the same network but with a different randomization due to biological noise (0.05).*



(a) Genomica (b) SAMBA (c) ARACNE

Figure 3.18: ***Impact of a multiple steep interactions in a linear chain on the performance of ARACNE, SAMBA and Genomica****. Impact on the performance (F-measure) of ARACNE, SAMBA Genomica of multiple steep interactions in a linear chain of 25 single-input single-output genes with linear interactions. The plot shows a series of boxplots for each position in the chain in which all linear interactions until that position are replaced by steep interactions. Each of the boxplots is the summary of 50 independent runs with the same network but with a different randomization due to biological noise (0.05).*

genes upstream of the steep interaction tending to be grouped together, as well as the downstream genes. On the other hand, when the steep interaction is close to the ends of the chain generally one big module is found. When multiple steep interactions are present, the F-measure decreases with an increasing number of steep interactions (Figure 3.18a) and saturates around 0.6. It becomes increasingly difficult to detect the relationship between all the genes in the chain.

When only linear interactions are present, ARACNE nearly always infers the chain topology perfectly, but shows a very steep decrease in F-measure with multiple steep interactions in the beginning of the chain (Figures 3.17c and 3.18c). The variation in F-measure is large for a nonzero number of steep interactions and decreases as number of steep interactions increases. A possible reason is that the mutual information between two genes downstream of the steep interactions on average decreases for a higher number of steep interactions.

This experiment shows that highly non-linear interactions do act as a buffer, masking the activity of downstream genes. The effect of this masking is highly dependent on the type of algorithm and the output that it generates. The high variation in performance may be due to sampling effects.

3.9 Summary

We have shown that the application of synthetic data on inference algorithms provides insight into the relation between different properties of the simulated model and the quality of the inferred network. Three different types of inference algorithms were tested, each of which exhibited different behavior to varying parameters of the synthetic data. The properties that were tested were network size, network topology, type and degree of noise, availability of expression data and interaction types between genes.

Experiments show that inference performance drops as network size increases, even if enormous amounts of data are available. This decay however is not as drastic as what might be expected. In order to infer larger networks with high confidence it seems that complementary approaches are needed such as adding additional data sources or incorporating domain knowledge. The topology of the network can have a strong impact on the performance of an inference algorithm, which should be taken into account when evaluating inference algorithms using synthetic datasets. It is encouraging to note that for two of the algorithms tested in this experiment, Genomica and ARACNE, the inference results of topologies that are known to be biologically more plausible, are better.

A clear effect of different types of noise on the inference performance of all algorithms is observed. Experiments show that noise is an important, and sometimes even a required factor during inference. All three noise types available in the SynTReN generator provoked a qualitatively different inference behavior, which supports their adoption in generators for synthetic data in general. We observed a substantial but decreasing benefit of supplying more expression datasets when trying to infer interaction networks. The algorithms tested behave differently, sometimes reaching a maximum performance where adding more arrays becomes pointless. Reaching the inference quality plateau

requires enormous amounts of data relative to the size of the inferred network.

The results show the added value of synthetic data in revealing operational characteristics of inference algorithms which are unlikely to be discovered by means of biological micro-array data, and thereby make a strong case for computer models of biological systems in leveraging systems biology research.

Chapter 4

Probabilistic Relational Models

4.1 Introduction

With the increased storage and processing capacity of current computers and the advent of huge online databases with interlinked information, an explosion in available data has occurred. Many of these datasets are stored in complex relational databases and most machine learning algorithms such as Bayesian networks [126], k-means clustering [106], decision trees [132] or neural networks [22] cannot be applied directly to such relational datasets since they are learned from data that is represented by a single table, called *attribute-value data*.

More expressive machine learning techniques that can represent both the variables and the relations that hold between them are called *relational data mining* methods. Some of the basic concepts of relational data mining were already outlined in 1970 by Plotkin [130] in the context of logical learning and the inductive logic programming community (ILP) has focussed on learning deterministic rules from relational data with successful applications in e.g. drug design and analysis of chemical databases. While relational data mining methods exist already for several decades, most machine applications are currently still applied to attribute-value data.

In the past 10 years a renewed interest in relational data mining methods has emerged with the availability of large relational datasets that, together with the increase in computer power, enabled efficient learning of models for these datasets. The domains of machine learning and ILP begun to incorporate the respective complementary aspects of the other domain, namely probabilistic representations to ILP [40] and relational extensions to machine learning

[55, 121]. The domain that now covers these two communities is called *statistical relational learning* (SRL) [39, 60], sometimes also referred as *probabilistic logic learning* or *probabilistic inductive logic programming*, and its research area focuses on learning in relational data with a strong probabilistic structure. SRL lies at the intersection of machine learning, knowledge representation and uncertainty reasoning.

4.2 Statistical relational learning

The two most common formalisms for representing SRL systems are based on either logic or frame-based formalisms and the probabilistic aspects are usually based on graphical models [84] or stochastic grammars. Many SRL approaches are defined on graphical models and often on directed models such as Bayesian networks (BN) [83, 126] or Probabilistic Relational Models (PRMs) [55, 61, 94]. However there is recently a growing interest in the application of undirected models such as Markov networks (MNs).

Graphical models are a widely used technique that have been applied to a variety of machine learning problems. They provide an intuitive way of representing complex models as a composition of simple relations between its constituents. One of the most important and widely known types of graphical models is a Bayesian network. Bayesian networks have been applied in various research areas, ranging from forecasting and classification to diagnostics and computer vision applications. Despite their elegance and widespread application, Bayesian networks have one major limitation which is that they can only be applied to attribute-value data and therefore not directly to relational datasets. Other types of graphical models include for example Markov models and Kalman filters.

From both SRL subfields, different approaches are proposed to bridge the gaps between the different techniques. A first class of models for density estimation in relational data sets are *probabilistic relational models* (PRMs) [55, 61, 94]. PRMs extend Bayesian networks to the relational domain and similarly relational Markov networks (RMNs) are the relational extension to undirected Markov networks. Contrary to PRMs, RMNs are not limited by any acyclicity constraints and can therefore represent arbitrary relations between variables, for example autocorrelation. Models for datasets with many autocorrelation dependencies can often not be structured in an acyclic manner and thus PRMs cannot be applied directly. However, the higher representational capabilities of RMNs usually come with an increase in computational cost for learning cyclic models which becomes prohibitively high for large cyclic graphs.

Another variant of this class of models are *relational dependency networks* (RDNs) that combine some of the advantages of both PRMs and RMNs. They have both the ability to represent and learn cyclic dependencies and to employ

efficient learning techniques. The main difference between RDNs and other probabilistic models such as directed PRMs and relational Markov networks is that RDNs do not specify a single joint distribution but rather a set of conditional probability distributions that are learned independently. This implies that the learned model is only an approximation of the true joint probability distribution. However, in some practical applications, RDNs have proven to approximate the true joint probability distribution quite well.

A second class of models for density estimation extend logic programming towards probabilistic reasoning. This class of models is called *probabilistic logic models* (PLMs). PLMs specify a probability distribution over all possible truth assignments to the groundings of the first-order formulae and include for example Bayesian logic programs (BLPs) that augment the power of Bayesian networks with logic programs [39, 88], Markov logic networks (MLNs) [137] and stochastic logic programs (for an overview of these techniques, see [41]).

For some undirected models such as Markov models and hidden Markov models, inference can be performed efficiently. The extensions of Markov models to use relations and logic, have led to relational Markov models and logical hidden Markov models. While less expressive than some more general approaches, these techniques are in principle more efficient and the existing learning algorithms for their non-relational counterparts can be applied almost directly.

In this Chapter, we will focus on one particular class of these SRL models, namely Probabilistic Relational Models [55, 61, 94]. PRMs have been applied to a variety of relational machine learning problems [34, 62, 122] and several applications [144, 145, 146, 147] were developed by E. Segal in the domain of bioinformatics. PRMs offer an elegant way for describing a biclustering model that is easily extensible towards integrating additional data sources as will be discussed in Chapter 5.

As more complex models will be developed, PRMs may at some point prove to be too constrained by the acyclicity constraints and the limitation to only directed graphs. More expressive models such as Bayesian logic programming [88], Markov logic [137] or the application of undirected models such as RMNs may provide interesting research tracks in these cases.

4.3 Bayesian networks

As mentioned above, a *Bayesian network* [83, 126] is one of the most commonly used types of graphical models. We will present an overview of the main concepts of Bayesian networks and highlight some properties that will be important when we introduce PRMs in Section 4.4.

Consider a set of n random variables $X = \{X_1, \ldots, X_n\}$. An element $x \in X$

specifies a point in the space that is defined by the Cartesian product of all variables X_i. A *joint probability distribution* (JPD) can now be defined over all points x by assigning a probability to each of these points. The most straightforward approach to accomplish this is by means of a table that lists all individual points and that assigns a probability to each of them. However, the size of such a table is exponential in the number of variables n. In practice, there are often independencies and conditional independencies between many of the variables X_i. Taking these into account leads to a much more compact representation of the JPD. One of the most often used techniques to accomplish such a compact representation is by means of a Bayesian network.

A Bayesian network describes the probabilistic relationships that exist between the random variables X_i and consists of two parts: (1) a *directed acyclic graph* (DAG) representing the independency relations between the random variables X_i where each random variable X_i corresponds to a node in the graph and a directed arc in the DAG represents a parent-to-child relationship between two variables; (2) the second component is a *conditional probability distribution* (CPD) for each of the variables. Together these components describe the joint probability distribution (JPD) $P(X_1, ..., X_n|\theta, S_{DAG})$ over all the variables of the Bayesian network. The CPDs are parameterized by a set of parameters θ and the DAG structure is represented by S_{DAG}.

Two sets of variables X and Y in this graph are said to be *conditionally independent* given a third variable Z if $P(X, Y|Z) = P(X|Z)P(Y|Z)$. Each node X_i in the graph is conditionally independent of any subset A of nodes that are non-descendants of X_i given a joint state of the parents $Pa(X_i)$ of X_i, namely: $P(X_i|A, Pa(X_i)) = P(X_i|Pa(X_i))$. This allows us to write the JPD as a product of conditional probability distributions:

$$P(X_1, ..., X_n|\theta, S_{DAG}) = \prod_{i=1}^{n} P(X_i|Pa(X_i), \theta, S_{DAG}) \tag{4.1}$$

Figure 4.1 illustrates a classical example of a Bayesian network. This example contains three variables *Rain*, *Sprinkler* and *GrassWet* that represent if it is raining, if the sprinklers are on and if the grass is wet respectively. The model assumes that *Sprinkler* depends on *Rain* and that *GrassWet* depends on both *Sprinkler* and *Rain*. The JPD that describes these dependencies is:

$$P(G, R, S) = P(G|R, S).P(S|R).P(R) \tag{4.2}$$

where the explicit dependency on the parameters θ and the dependency structure S_{DAG} have been omitted for notational convenience.

Usually, each conditional probability distribution $P(X_i|Pa(X_i), \theta, S_{DAG})$ depends on its own set of parameters θ_i, such that θ can be written as $\theta = \{\theta_1, \ldots, \theta_n\}$. This is also the case in Figure 4.1, where each variable has an associated table CPD with its own parameters. Assuming this independence between CPD

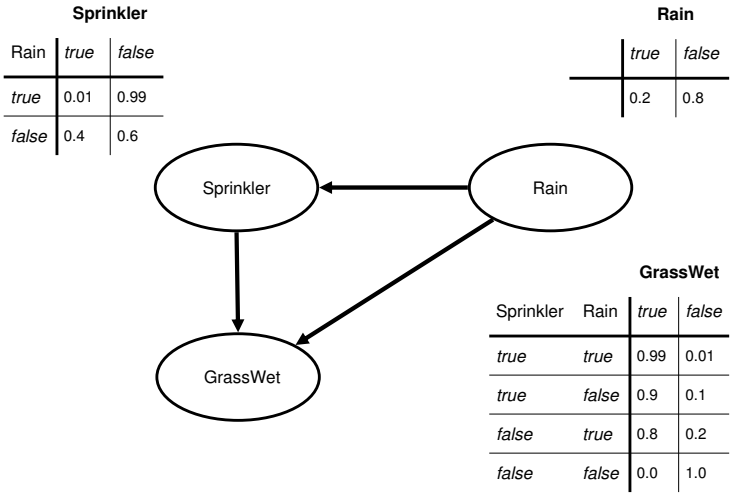


Figure 4.1: ***A simple Bayesian network, illustrating the basic concepts of Bayesian networks****. The network has three variables Rain, Sprinkler and GrassWet. The dependency arcs indicate that GrassWet depends on the state of Rain and Sprinklers and that Sprinklers depends on Rain.*

parameter sets, Equation 4.1 can be written as:

$$P(X_1, ..., X_n|\theta, S_{DAG}) = \prod_{i=1}^{n} P(X_i|Pa(X_i), \theta_i, S_{DAG}) \tag{4.3}$$

where $P(X_i|Pa(X_i), \theta_i, S_{DAG})$ is now a *local* distribution function for X_i.

The *Markov blanket* (MB) of a variable X_i is the set of variables that effectively shield off this variable from the rest of the network: given the values of variables in $MB(X_i)$, the probability distribution of X_i is conditionally independent of all other variables in the network. More formally: $P(X_i|MB(X_i), Y) = P(X_i|MB(X_i))$ for every set of variables $Y = \{Y_1, \ldots, Y_n\}$. One can prove that the Markov blanket of a variable X_i is the set of variables consisting of the parents of X_i, the children of X_i and the parents of the children of X_i.

4.3.1 Bayes theorem

In its most general form, Bayes' theorem defines the rule for updating belief in a certain hypothesis H given (additional) evidence E through the following equation, in which the explicit dependence on background knowledge is

omitted:

$$P(H|E) = \frac{P(H).P(E|H)}{P(E)} \tag{4.4}$$

$P(H|E)$ is called the posterior probability, $P(H)$ is the prior probability of the hypothesis H and $P(E|H)$ is called the likelihood, it gives the probability of the evidence assuming the hypothesis H is true. The denominator $P(E)$ can be regarded as a normalizing constant since it is independent of the hypothesis H.

In Bayesian modeling, the hypothesis is often defined as the set of parameters θ of a particular model and the additional evidence E is often the dataset that has been collected (represented by the random variable X). Redefining Bayes' theorem in function of these variables leads to Equation 4.5 with likelihood $L(\theta|X)$ and prior $P(\theta)$.

$$\begin{aligned} P(\theta|X) &= \frac{P(X|\theta).P(\theta)}{P(X)} && (4.5)\\ &\propto P(X|\theta).P(\theta) && (4.6)\\ &\propto L(\theta|X).P(\theta) && (4.7) \end{aligned}$$

4.3.2 Learning Bayesian networks

The parameters of a Bayesian network can be learned from an attribute-value dataset. When the structure of the network is known, two cases can be distinguished: learning from complete data and learning from incomplete data. If the structure of the network is unknown, structure learning approaches [30, 32, 71] are used to derive the dependency structure from the available data. For the models presented in this thesis, we assume a known structure and focus on the methods for parameter learning in case of complete and incomplete data.

Complete data

The goal of the learning procedure is to identify the set of parameters θ that maximize some scoring function. When no prior information is available for the parameters, the *likelihood function* is often used as a scoring function. However, the likelihood function is known to overfit the training data and in cases where prior information is available for the parameters in the form of a distribution $P(\theta)$, the posterior distribution can be used as a scoring function. The parameters associated with the maxima of these functions are called the maximum likelihood (ML) and maximum a posteriori (MAP) solutions respectively.

$$\begin{aligned} \hat{\theta}_{ML}(x) &= argmax_\theta P(\mathrm{X}|\theta, S) && (4.8)\\ \hat{\theta}_{MAP}(x) &= argmax_\theta P(\mathrm{X}|\theta, S).P(\theta) && (4.9) \end{aligned}$$

The identification of the maximum likelihood solution for the CPDs can be performed very efficiently [120], since Equation 4.8 decomposes into a set of n separate optimizations for each parameter set θ_i:

$$\hat{\theta}_{ML}(x) = argmax_{\theta} P(X|\theta, S) \tag{4.10}$$

$$= argmax_{\theta} \prod_{i=1}^{n} P(X_i|Pa(X_i), \theta_i, S) \tag{4.11}$$

Usually, the prior distribution $P(\theta)$ is chosen such that a similar decomposition applies to the prior (see Section 4.3.3). The MAP optimization problem is then reduced analogously to a set of n independent optimization problems.

Incomplete data

In case of incomplete data, the scoring function is maximized over all possible completions of the hidden variables H (where $H \subset X$). This leads to the following equation for the maximum likelihood solution:

$$\hat{\theta}_{ML}(x) = argmax_{\theta} \sum_{h \in H} P(X|\theta, S, H = h) P(H = h|\theta, S) \tag{4.12}$$

In case of incomplete data there is no longer an analogous decomposition of the scoring function similar to Equation 4.10 and identifying a global maximum is now a hard problem. Several approaches can be used to identify a *local* optimum of the scoring function, such as gradient descent algorithms, simulated annealing or MCMC strategies. An approach that has been specifically designed for optimizing likelihood functions, is the *Expectation-Maximization* algorithm [43]. In Section 4.4.5, the Expectation-Maximization algorithm will be introduced together with an illustrative example for probabilistic relational models.

4.3.3 Prior distributions

The prior distribution $P(\theta)$ often adheres to two assumptions that allow the posterior distribution to be decomposed in a set of factors that each can independently be optimized. These assumptions therefore often lead to significant decreases in computational cost for both inference and parameter learning tasks. The *global parameter independence* is a standard assumption on the form of the prior that is commonly used in Bayesian networks learning [71]. The assumption states that the prior over the parameters for the different attributes are independent. A second assumption that is often used is the *local parameter independence*, stating that the prior distribution in a particular node is independent from the different parent values of that node.

In Bayesian probability theory, *conjugate priors* are frequently used as particular form of prior. A probability distribution $P(\theta)$ is said to be *conjugate* to a

likelihood function $P(X|\theta)$ if their resulting posterior distribution is again a distribution from the same family as the prior. For example, the exponential family (of which the Gaussian distribution is a member) is conjugate to itself. Combining a Gaussian likelihood function with a Gaussian prior results again in a Gaussian posterior. Conjugate priors allow compact algebraic notations for the posterior and are therefore a convenient way to define prior knowledge. However, despite their algebraic convenience, conjugate priors do not always optimally reflect prior domain knowledge and sometimes more complex prior distributions are required.

4.3.4 Conditional probability distributions

Table CPD

CPDs can be represented in many different ways. One of the most straightforward CPDs is a table CPD which describes the relation between a discrete variable and its discrete parents. In a table CPD, a separate parameter for each combination of parent values u, describes the probability $P(X_i|Pa(X_i) = u)$. For example, in the case of two parents each with binary values, a table CPD for a binary variable is defined by $2^2 = 4$ parameters. An example of a table CPD for a binary node with two binary parents is given in Table 4.1.

x	**y**	$P(Z = 0\|X = x, Y = y)$
0	0	θ_{00}
0	1	θ_{01}
1	0	θ_{10}
1	1	θ_{11}

Table 4.1: Example of a table CPD for a binary node Z with 2 binary parents X and Y. This table CPD has four parameters $\theta_{i,j}$. Note that $P(Z = 1|X = x, Y = y)$ is defined implicitly by the relation $P(Z = 0|X = x, Y = y) + P(Z = 1|X = x, Y = y) = 1$.

Gaussian CPD

One of the most commonly used CPD for continuous data is a Gaussian or Normal distribution. Many variables are approximately normal or log-normal distributed in the biological domain, making a Normal distribution a straightforward choice in such cases. If the underlying variable is not normally distributed, alternative choices such as Poisson, Gamma or uniform distributions can be used depending on the case. A Normal CPD for a single variable X is defined as:

$$P(X = x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp[-\frac{(x-\mu)^2}{2\sigma^2}] \tag{4.13}$$

where the parameters μ and σ are the mean and standard deviation of the Normal distribution.

4.4 Probabilistic Relational Models

4.4.1 Introduction

Probabilistic Relational Models (PRMs) [55, 61, 94] were recently developed as an extension of Bayesian networks to the relational domain. PRMs have since been successfully applied in wide range of settings such as hypertext classification [62], collaborative filtering [122], ecosystem analysis [34] and genomics [144, 145, 146, 147]. In this Chapter we introduce PRMs with an illustrative example. The key language concepts and a notation are defined, which will be used consistently throughout this dissertation.

4.4.2 Definitions and notation

A PRM specifies a probability distribution over a relational dataset and consists of two main parts: (1) a relational component that describes the relations in our domain and (2) a probabilistic component that describes the probabilistic dependencies between the different entities in the domain. Given a particular dataset instance, a PRM specifies a joint probability distribution over this particular instance. The notation used in this thesis is largely consistent with the notation as defined in Segal [143], with some minor deviations to reduce notational overhead.

Figure 4.2 shows a fictitious example that will be used throughout the remainder of this Chapter to illustrate various aspects of PRMs. The PRM models a patient/treatment dataset containing the results of some (fictitious) *Influenza* infections. In the proposed model in Figure 4.2, we see that the patient's degree of illness depends on the type of virus and its pandemic severity index, on the drug class of the administered drug and on the patient's age.

For each PRM, a *relational schema* Σ exists, that defines a set of *classes* $\mathcal{C} = C_1, ..., C_n$ and the *relations* that hold between these classes. If we draw an analogy between PRM concepts and relational database concepts, then a relational schema relates to the database structure and a class relates to PRMs much like a table relates to a relational database. Table 4.2 shows all related concepts between PRM language and relational database language. In our example, the PRM classes are *Patient*, *Drug* and *Influenza virus*.

Each of these classes has a set of descriptive *attributes*, an attribute A of a class C is denoted as $C.A$ and the set of attributes of a class C is $\mathcal{A}[C]$. Each attribute can take a particular value from a predefined value space $Val(C.A)$, which can be a discrete set or a continuous range of values. For example, the *Influenza_virus* class has five attributes: *type*, *subtype*, *host*, and *pandemic SI*. The value space for *type* is for example: $Val(Influenza.type) = \{A, B, C\}$.

In addition to a set of attributes, each class can have a number of *reference slots*, which indicate the relations between the different classes. We use the notation

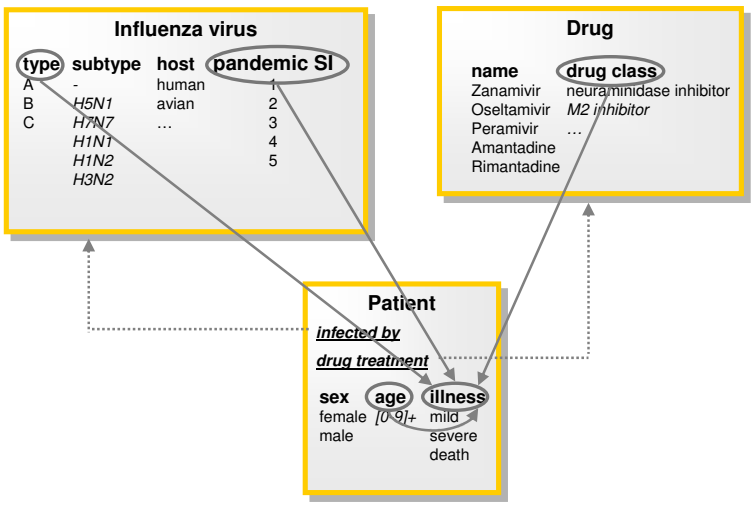


Figure 4.2: ***Relational dataset and PRM model of a fictitious Influenza infection dataset, illustrating the key concepts of PRMs.*** *A patient/treatment model is shown, in which the impact of both the* Influenza *virus type and the type of drug treatment are linked to the degree of illness of the patient. The class* ***Influenza virus*** *contains the following attributes:* type *and* subtype*: define the type of Influenza strain; host: the host organism that the virus uses to replicate;* pandemic SI*: the pandemic severity index, '1' indicates low severity and '5' high severity. The class* ***Drug*** *has attributes* name *and* drug class *referring to the name of the drug and the classification of the drug. Finally, the class* ***Patient*** *contains attributes* sex, age *and* illness*. Next to the attributes,* Patient *also has two reference slots* infected by *and* drug treatment *that refer to a particular* Influenza virus *object and* Drug *object respectively.*

$C.\rho$ to indicate a reference slot ρ of class C and the complete set of reference slots for a particular class C is denoted as $\mathcal{R}[C]$. Two concepts are associated with a reference slot $C.\rho$: the *domain type* $Dom[\rho]$ which is the class C and the *range type* $Range[\rho]$ which is the class to which $C.\rho$ refers to. Applied to our example, the class *Patient* has two reference slots: *Patient.infected_with* and *Patient.treatment* which respectively relate *Patient* to the classes *Influenza_virus* and *Drug*. So the domain type for $infected\_by$ is $Dom[infected\_by] = Patient$ and the range type is $Range[infected\_by] = Influenza\_virus$.

Reference slots can be linked together to form a *slot chain*. This is formally as a sequence of reference slots $\bar{\rho} = (\rho_1, ..., \rho_n)$ for which: $Range[\rho_i] = Dom[\rho_{i+1}], \forall i$. For example, *Patient.infected_by.Influenza_virus* indicates the (set of) virus(es) with which the patient was infected with.

Once a relational schema is defined, a specific *instance* $\mathfrak{I}$ can be created of that schema. An instance defines the set of objects c for each class C: $\mathfrak{I}[C]$ and for each object c it defines a value for each attribute $c.a$ ($\mathfrak{I}[c.a]$) and also a value for each reference slot $c.\rho$ of that object. Figure 4.3 shows an example of an

PRM concept	relational database concept
relational schema	database scheme
class	table
attribute	column
reference slot	foreign key
instance	database instance with values for each item
skeleton	database filled (with all null values)

Table 4.2: Relationships between PRM concepts and their counterparts in relational database modeling.

instance (Figure 4.3b) for a particular schema (Figure 4.3a).

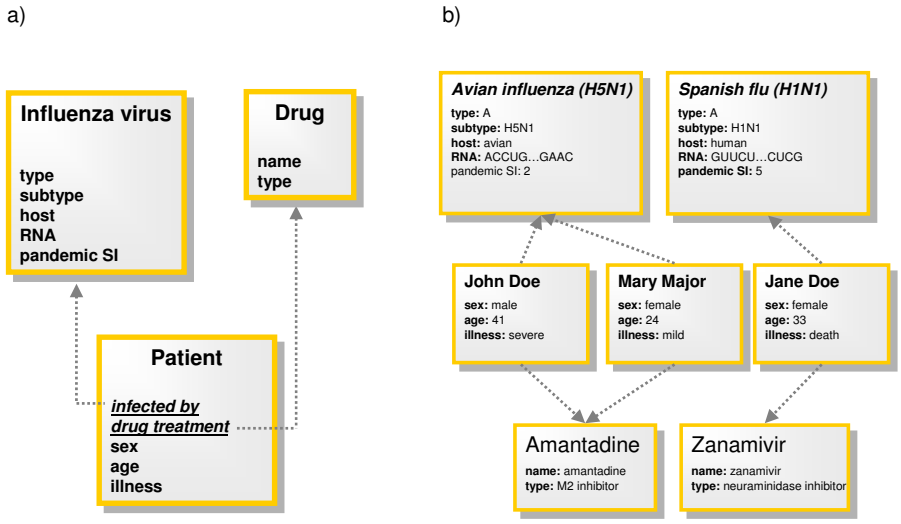


Figure 4.3: ***Relational schema and example instance of a relational schema.*** ***(a)*** *Relational schema for the example in Figure 4.2. Rectangles are the classes, dashed lines are the relations between the classes.* ***(b)*** *Example instance for the relational schema in (a). Rectangles are objects of a specific class and dashed lines indicate the relation between two specific objects.*

We can now define a *relational skeleton* σ_r of a relational schema as a partial instantiation of an instance: the set of objects and their relations are defined by the skeleton, but the values of the attributes remain unspecified.

Finally, the *set of parents* of an attribute $C.A$ is defined as $Pa(C.A) = \{U_1, \ldots, U_p\}$, where U_i is either of the form $C.B$ or $C_y.\bar{\rho}.B$ (where B is the parent attribute and $\bar{\rho}$ is a slot chain). The parents of a class attribute $Pa(C.A)$ are *formal parents*: each object c of a class C has a different instantiation of parents, depending on the instance $\Im$. A parent can be a *direct parent* of the form $c.B$.

For example *Patient.age* is a direct parent of *Patient.illness* in our example of Figure 4.2. Parents can also be *indirect* by means of a slot chain. For example, *Patient.treatment.type* is an indirect parent of *Patient.illness*.

Having introduced the key concepts in PRM language, a PRM can now be defined as a model that specifies a probability distribution over all possible completions $\Im$ of the skeleton σ_r. More formally:

Definition 1 (Probabilistic Relational Model (PRM) Π) *For a given relational schema Σ, for each class $C \in \mathcal{C}$ and for each attribute $A \in \mathcal{A}[C]$, a PRM defines a valid conditional probability distribution (CPD) $P(\, C.A \,|\, Pa(C.A)\,)$.*

4.4.3 Joint probability distribution

Definition 2 (ground Bayesian network (GBN)) *A ground Bayesian network is defined by a relational skeleton σ_r together with a PRM Π, in the following way:*

- *For every attribute A of every object c of each class C in σ_r, there is a corresponding node $c.A$ in the ground Bayesian network.*
- *Each node $c.A$ depends probabilistically on its parents $Pa(c.A)$ according to the conditional probability distribution $P(\, C.A \,|\, Pa(C.A)\,)$*

Figure 4.4 shows the ground Bayesian network for our running example. Similarly to the chain rule in Bayesian networks, the joint probability distribution of the ground Bayesian network is defined as the product over all conditional probabilities $P(\, x \,|\, Pa(x)\,)$ of each node x in the GBN. This means taking the product over all classes C, over all objects c of each class and over all attributes A of each object of the probability $P(\, c.A \,|\, Pa(c.A)\,)$. Like Bayesian networks, the ground Bayesian network is also required to be acyclic. A sufficient condition to achieve this is that the dependency relations in the PRM are acyclic. The *Pro*Bic model in Chapter 5 adheres to this condition.

Definition 3 (Joint probability distribution (JPD)) *:*
For a given instance $\Im$, the joint probability distribution of the ground Bayesian network associated with the instance $\Im$ and the relational skeleton σ_r, is given by:

$$P(\Im|\sigma_r, S, \theta_S) = \prod_{C \in \mathfrak{C}} \prod_{A \in Attr(C)} \prod_{c \in \sigma_r(C)} P(\, \Im[c.A] \,|\, \Im[Pa(c.A)]\,) \tag{4.14}$$

where S represents the dependency structure of a PRM model and θ_S the parameters associated with that particular dependency structure S. The JPD associates a probability to all possible instances that are a completion of the relational skeleton σ_r.

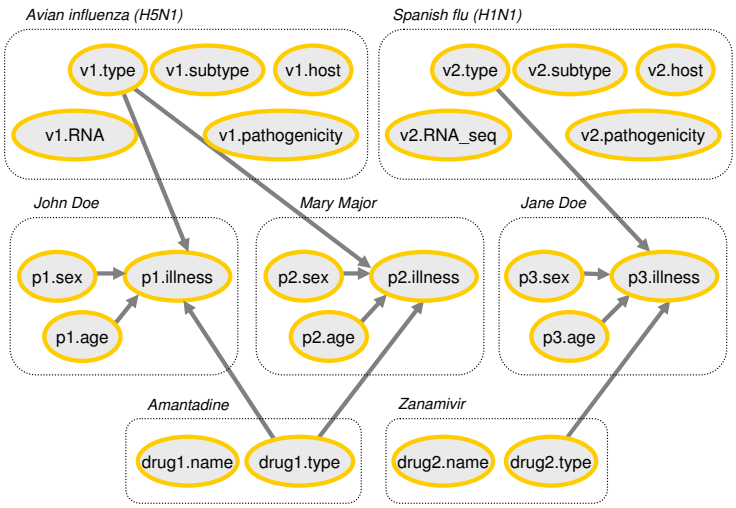


Figure 4.4: ***Ground Bayesian network for the example in Figure 4.2.***

4.4.4 Inference

The JPD associates a probability to all possible instances that are a completion of the relational skeleton σ_r. One of the main questions to be answered is what the most likely model is that best explains the collected data. This question can be addressed using the ML or MAP principles that identify the model with the highest likelihood (or posterior) given the data: $argmax_M P(M|D)$

Various other questions can also be formulated as a Bayesian inference problem. Applied to our running example (Figure 4.2), we can for example ask the following kinds of questions:

- What is the probability that a patient who is infected with *Influenza* subtype *H5N1* and is being treated with an M2 inhibitor, will die?
- Do age and sex of the patient influence its clinical outcome? If so, is this true for all the drugs and virus strains?
- Is there a difference in clinical outcome between the two drug classes *neuraminidase inhibitor* and *M2 inhibitor*?
- Can different groups of patients be identified for which the treatment works differently (personalized medicine)?

Most of these questions can be equally well addressed in separate models using various other techniques. However, PRMs offer a *single* unified model of the complete domain where each of these questions can be answered in a probabilistic way.

4.4.5 Learning PRMs

In order to answer these inference questions for the PRM, first a model needs to be learned based on the available data. In the most straightforward way, this involves learning the parameters of the model, given that we have complete data. Most biologically interesting problems however lead to a model with hidden attributes, and therefore lead to learning problems with incomplete data. For example, our *Influenza* example in Figure 4.5, the *Patient.group* attribute is initially unknown and will be learned from the data.

In principle any general learning technique that is applicable to Bayesian networks can also be applied to PRMs as a PRM implicitly describes a Bayesian network. However, the underlying Bayesian network is often impractically large, leading to intractable calculations. For most classical Bayesian inference approaches, a modified procedure can be applied to PRMs that takes into account the specific relational structure which is present in the PRM. Structured exact inference [93, 128, 94] is such a method, but it leads however to intractable calculations for most biological applications.

Approximate inference methods are better suited to biological problems. For example, Markov chain Monte Carlo (MCMC) approaches and more specifically Metropolis-Hastings and Gibbs sampling strategies have been applied in the context of PRMs [158]. While the Gibbs sampling method is guaranteed to converge (for an infinite number of samples) to the full distribution if certain conditions are met, in practice a finite number of samples are often collected around one or more local optima. Alternative approaches such as variational approaches [73] and blocked Gibbs sampling have been used to address this problem, but with varying degrees of success. One of the standard approaches for parameter estimation in the case of *hidden* (or *latent*) variables is *Expectation-Maximization* (EM) [43]. This technique exists in many different shapes: *soft assignment* and *hard assignment* EM [87], generalized EM approaches,... As we will show in Chapter 5, EM leads to excellent decomposition and fairly good convergence properties for PRMs if certain conditions are met.

In the following sections we will briefly summarize the general parameter learning approaches outlined in [55, 61] for the complete data case and for the extension towards incomplete data [143]. Structure learning involves learning the structure itself of the dependency network in PRMs and it is not covered in this thesis. We refer to [55, 61, 143] for description of structure learning techniques in PRMs.

Parameter learning

Similar to parameter learning in Bayesian networks, a full Bayesian approach to parameter learning for PRMs involves finding the joint probability distribution over all possible combinations of parameters values. For most biological problems of interest, such a full Bayesian approach is computationally in-

tractable. Fortunately, bioinformaticians are usually not interested in the full distribution, but rather in the *maximum likelihood* solution(s). In almost all practical situations, the search for such a ML or MAP solution can be performed with a significantly reduced computational cost.

Maximum likelihood parameter estimation in case of complete data
Parameter learning in the complete data case is relatively straightforward. The structure S of the PRM is fixed in this setting and there are no unknown attribute values. Our starting point for PRMs is the likelihood function of the parameter set θ_S:

$$l(\theta_S|\Im, \sigma, S) = P(\Im|\sigma, S, \theta_S) \tag{4.15}$$

$$= \prod_{C \in \mathfrak{C}} \prod_{A \in Attr(C)} \prod_{c \in \sigma_r(C)} P(\,\Im[c.A]\,|\,\Im[Pa(c.A)]\,) \tag{4.16}$$

This function is in fact the likelihood function of the ground Bayesian network that is associated with the PRM. The difference of the PRM log-likelihood function compared to that of Bayesian networks, is that the parameters of nodes in the ground Bayesian network that are associated with the same attribute, are *shared*, meaning they are forced to be identical.

The goal of maximum likelihood parameter estimation is to find the set of parameters θ_S that maximize the likelihood function, given $\Im$, σ and S. Similar to parameter learning in Bayesian networks [71], the maximum likelihood solution for PRMs can also be independently calculated for the CPD of each attribute. Analogously to parameter learning in Bayesian networks (Section 4.3.2), prior distributions can be defined over the parameters and the posterior is maximized instead of the likelihood function.

Maximum likelihood parameter estimation in case of incomplete data
To illustrate parameter estimation for the incomplete data case, we extend the *Influenza* example of Figure 4.2 with an additional categoric attribute *Patient.group* whose value is initially unknown for each of the patients (i.e. it is a *hidden attribute*). The attribute *Patient.group* clusters patients in mutually exclusive groups and indicates how each patient responds to a treatment given his/her underlying genetic background. It is conditionally dependent on the value of another attribute *Patient.genome*, containing (part of) the DNA sequence of that patient's genome. The dependency relation is not further specified in this fictitious example, but one could for example use a multinomial model indicating the presence or absence of some particular genes that are associated with the treatment response. By assigning patients to a particular group that is linked to a genotypic profile, the model thus identifies patient groups with a similar genotypic profile that respond similarly to a particular treatment. Figure 4.5 illustrates this extended model.

In this extended example we need to learn both the model parameters and the hidden attribute values. The set of attributes with missing values is denoted

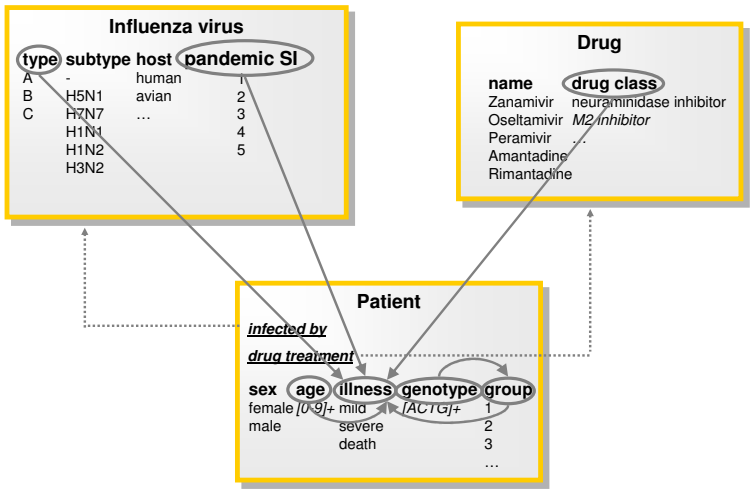


Figure 4.5: ***An extension of the patient/treatment model with an additional hidden attribute*** *Patient.group. The attribute indicates the difference in response that each patient will have for a treatment due to his/her genetic background, by grouping patients together with the same genotypic profile and which respond similar under similar conditions of infection and treatment.*

H. To obtain the complete likelihood function, a value needs to be calculated for every possible completion of missing attributes H. The distribution over all possible completions is then used to estimate the model parameters, as illustrated in Equation 4.17.

$$l(\theta_S|\Im, \sigma, S) = \sum_{h \in H} P(\Im|\sigma, S, \theta_S, H = h) \cdot P(H = h|\sigma, S, \theta_S) \qquad (4.17)$$

Contrary to the complete data case, the likelihood function does not decompose into independent contributions in case of missing attributes. This type of inference is generally intractable, so we resort to optimization procedures that identify local maxima of this likelihood function. Several approaches can be applied to perform local function optimization such as Monte Carlo methods, Gaussian approximation and gradient-based approaches. See also Heckerman and Geiger [71] for a more detailed overview of learning methods for Bayesian networks.

One particularly interesting approach for PRMs is to use *Expectation-Maximization* (EM) [43] in case of incomplete data. EM is guaranteed to identify a local maximum of the likelihood function by applying the following procedure. Starting with an initial set of parameters values $\theta_S = \theta_S^{(0)}$ (which can be randomly chosen or derived through other methods), an *Expectation* step and a *Maximization* step are iterated until convergence.

In the Expectation-step, the full posterior *distribution* over the hidden attributes is computed, given the observed attributes and the current set of parameters $\theta_S^{(t)}$: $P(H|I, \sigma, S, \theta_S^{(t)})$. This computation can be done by performing inference in the ground Bayesian network.

In the Maximization-step, we maximize the log-likelihood function with respect to the parameter set θ given the computed distribution over the hidden attributes in the E-step. This step is similar to the maximum likelihood parameter estimation in case of complete data. Whereas in the complete data case the single value for that attribute would be used, now a *distribution* over the possible values of the hidden attribute is used.

The above approach is called *soft assignment EM*. Another variant is *hard assignment EM* where not the full distribution over all values of the hidden attribute is calculated in the E-step, but rather the maximum likelihood assignment is calculated:

$$h^* = argmax_{h \in H} P(H = h|I, \sigma, S, \theta_S^{(t)}) \tag{4.18}$$

The hard assignment variant can in some situations significantly reduce the computational cost, as we will see in Chapter 5. This hard assignment EM is guaranteed to converge to a local maximum of the likelihood of the completed data, $P(I, H = h|\sigma, S, \theta_S)$, where h represents a complete assignment to the missing attributes H.

4.5 Summary

We summarize the main concepts that were introduced in this Chapter and their relation to the rest of this dissertation.

- A brief introduction was given, relating Probabilistic Relational Models to graphical models and relational data mining.
- We introduced Bayesian networks and how Bayesian networks are learned in case of complete data and incomplete data. Two often used conditional probability distributions, namely table CPDs and Gaussian CPDs, were defined, allowing modeling of both discrete and continuous data.
- The larger part of this Chapter was concerned with the definition of PRMs and their relation to Bayesian networks. We defined a fictitious relational dataset regarding *Influenza* infections with a set of patients receiving different treatments and we showed how PRMs can be used to model such a relational domain in a probabilistic model. Learning PRMs in case of complete and incomplete data was related to learning in Bayesian networks and more specifically, the Expectation-Maximization algorithm was highlighted as an interesting learning technique for PRMs in case of incomplete data.

Chapter 5

*Pro*Bic model

Results in this Chapter were developed in collaboration with Hui Zhao (KULeuven).

5.1 Introduction

The identification of gene regulatory networks based on gene expression data is a highly active field of research [14, 54, 99, 145]. It is also an underconstrained problem as the number of possible interactions and the parameters governing their relations far exceeds the dimensionality of the available data. Moreover, current microarray gene expression data is inherently noisy, further obscuring a detailed view on the regulatory interactions.

Many techniques have therefore been developed to obtain robust representations of the underlying network by reducing the parameter space, often by grouping genes into *regulatory modules*. An overview of such techniques is given in Chapter 2. Most module learning methods have focused on the identification of a *global* set of modules, aiming to identify the high level modular structure of transcriptional regulation.

From a biologist perspective however, a highly focused *local* analysis around a specific pathway is often more interesting. A large number of bottom-up methods exist that allow a more detailed reconstruction of pathways, typically containing 5-100 genes. However, usually highly specific types of data need to be gathered by means of low throughput and time-consuming experiments to determine the individual interaction parameters in the pathway. The application of such methods on a genome wide scale is either infeasible or leads to models with large parameter uncertainties and/or multiple possible solutions.

In between these global and local module identification methods, there is also a need for methods that combine the advantages of both approaches. Such

method should have few requirements on the type and quality of the provided data. It should be able to perform locally directed module identification on specific genes or pathways of interest while also incorporating the knowledge derived from global module identification. Few algorithms bridge this gap, usually by allowing directed searches in gene expression data, namely ISA [19], GEMS [169], Gene Recommender [123] and QDB [47].

5.2 Biclustering

The term *biclustering* was first introduced by Cheng and Church [29] in the context of gene expression data. Biclustering algorithms perform simultaneous clustering in both the gene and the condition dimension. The result is a set of biclusters that each contain a subset of genes and conditions. Starting from a m by n data matrix E, with a set of rows $R = r_1, \ldots, r_m$ and a set of columns $C = c_1, \ldots, c_n$, a bicluster (I, J) can be defined as a subset of rows $I = i_1, \ldots, i_r$ and a subset of columns $J = j_1, \ldots, j_s$ of E, such that the bicluster satisfies some specific measure of homogeneity [110].

An excellent survey of the different classes of biclustering algorithms is given in Madeira and Oliveira [110], where biclusters are subdivided in a number of different categories: constant biclusters, biclusters of constant rows or constant columns, biclusters of additive coherent values, biclusters of multiplicative values, biclusters of coherent evolutions and biclusters of coherent sign changes. According to this categorization, *Pro*Bic aims to find biclusters with constant columns. An example of such biclusters is given in Figure 5.1. From a biological perspective, a bicluster with a constant columns model represents a set of genes that have a similar expression level (per condition) for a subset of conditions.

5.2.1 State of the art biclustering algorithms

In the domain of gene expression data analysis, a number of different biclustering algorithms have been developed since the first publication of Cheng and Church (CC) [29] in 2000. CC-biclustering aims to find biclusters with minimal *mean squared residue*, which is obtained if for example the bicluster has constant values or if the genes (conditions) in the bicluster are identical up to a constant row (column) term per bicluster. Yang et al. [173] further improved this algorithm and allowed for missing values in the expression matrix.

Plaid models, proposed by Lazzeroni et al. [98], are a statistical model in which the expression value in a bicluster is represented as the sum of the main effect, the gene effect, the condition effect, and a normally distributed noise term. An expression value in the overlap region of two biclusters is modeled as the *sum* of the individual bicluster contributions. Plaid models use a greedy strategy

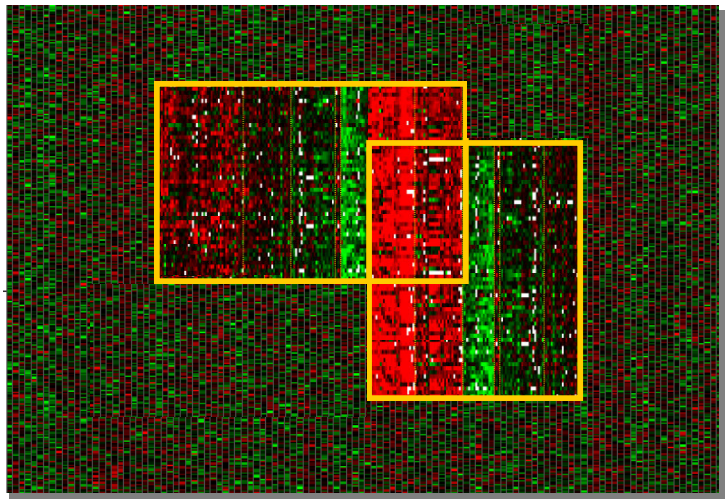


Figure 5.1: ***Illustration of two overlapping biclusters with constant columns.*** *The rows of the matrix represent genes and the columns represent conditions under which these genes were measured.*

where errors can accumulate and the biclusters tend to overlap easily, thereby creating large and biologically meaningless overlaps [64].

Ben-Dor et al. [17] attempted to identify order-preserving sub matrices (OPSMs). The method identifies biclusters in which the expression levels of the genes have an identical ordering for the subset of bicluster conditions. The main drawbacks of this method are the computational requirements for realistic datasets and the inability to accommodate for noise in the data (the method requires a perfect ordering for all genes).

Segal et al. [144] propose a gene clustering approach based on PRMs where the gene clusters can overlap. There is no clustering of the conditions in this model and the expression value in the overlap region is modeled as the sum of the individual cluster contributions.

The Iterative Signature Algorithm (ISA) [19] identifies single biclusters using an approach that is related to singular value decomposition (SVD). It tends to select strong biclusters many times (requiring masking of previously found biclusters to identify new biclusters) and it is sensitive to its parameter values. Variants of this algorithm have been published such as PISA [90] and USA [89].

Tanay et al. introduced SAMBA [154], a statistical approach to biclustering using a bipartite graph representation. SAMBA has a high specificity but

lower sensitivity for recovering biclusters [162] and has no explicit concept of overlapping biclusters.

Dhollander et al. applied a Bayesian approach to biclustering, QDB (query-driven biclustering [47]) that allows identification of specific biclusters of interest by using a set of *seed genes* that is a priori known or expected to be part of a bicluster. By means of a *resolution sweep*, the authors identify biclusters at different resolutions of interest.

Gu and Liu [64] also proposed a Bayesian approach to biclustering and compared their method to several other methods. While their model supports either genes or conditions to be part of multiple biclusters, a simultaneous overlap for both genes and conditions is not allowed.

5.2.2 Comparison *Pro*Bic vs. state of the art

Table 5.1 summarizes and compares the characteristics of *Pro*Bic with respect to the current state of the art algorithms. The *Pro*Bic biclustering model builds upon the expertise of previous work and extends the current generation of biclustering methods in the following areas:

- Many biclustering algorithms have no explicit model for the identification of multiple biclusters [17, 19, 29, 47, 173]. Heuristic approaches such as masking and the use of well chosen initializations, are often applied to identify more than one bicluster. *Pro*Bic accommodates for this and uses a model-based approach for identifying multiple and potentially overlapping biclusters simultaneously.
- Current methods, that allow to model multiple biclusters simultaneously [16, 98, 144], all have an additive model for overlapping biclusters. Computational tractability is assumedly the main reason for using an additive overlap model. *Pro*Bic can use different overlap models that still retain computational tractability. One specific implementation is given in Section 5.3.
- *Pro*Bic can be used both in a global mode as in a query-driven mode. Biclusters can thus both be identified by incorporating prior knowledge where applicable (in the form of a set of seed genes) and by using a global biclustering approach.
- *Pro*Bic allows for *diametrical biclustering*: genes with both correlated and anticorrelated profiles are grouped within the same bicluster. The explicit extension of one-dimensional diametrical clustering methods [46] towards the biclustering domain has to the authors knowledge not been previously published for non-constant biclusters[1].

[1] Wolf et al. [168] have published a biclustering method that identifies a constant bicluster that also includes the negative set of genes with the same but negated constant value.

- *Pro*Bic performs the biclustering efficiently by well-chosen decompositions of the posterior distribution (see Section 5.5). No prior discretization of the gene expression data is required and *Pro*Bic is able to remove seed genes from the resulting bicluster if that gene has no matching expression profile with the bicluster. This feature is important in practice as biologists often have a list of *potential* seed genes rather than a list of *definite* seed genes.

Name	GBCL	MULT	OVL	PRB	NMR	QD	DB
BBC [64]	+	+	-	+	+	-	-
ccc-biclustering [111]	+	+	-	-	-	-	-
cMonkey [135]	+	+/-[1]	-	+	+	-	-
DISTILLER [102]	+	+	-	-	+/-[2]	-	-
GEMS [169]	+	-	-	+	+	-	-
ISA [19]	+	-	-	-	+	+	-
OPSM [17]	+	+	-	-	-	-	-
Plaid models [98]	+	+	+	-	-	-	+
*Pro*Bic	+	+	+	+	+	+	+
QDB [47]	-	-	-	+	+	+	-
ReMoDiscovery [37]	+	+	-	-	+/-[2]	-	-
SAMBA [154]	+	+	-	-	+	-	-
SBK [16]	+/-[3]	+	+	+	+	-	-

[1] cMonkey also tends to repeatedly find the same biclusters from different seeds, they have however integrated a prior in their model that preferably includes each gene in two biclusters.
[2] ReMoDiscovery is two-phased. In the first phase of the algorithm, seed modules are identified by means of a non-robust Apriori-based search step. In the second phase these modules are expanded using a noise robust expansion step.
[3] The SBK algorithm is in between a gene clustering and a biclustering algorithm. It identifies gene clusters with an associated regulatory program that clusters *all* the conditions for that gene cluster.

Table 5.1: Qualitative comparison table of biclustering algorithms for gene expression data. Each of the following columns highlights a separate aspect: **GBCL** (global biclustering)*: indicates if the algorithm performs a global biclustering;* **MULT** (multiple biclusters)*: indicates if the model can identify multiple biclusters simultaneously (without masking or other heuristics);* **OVL** (overlap)*: indicates if overlapping biclusters are modeled with an explicit overlap model;* **PRB** (probabilistic)*: indicates if the algorithm is based on a probabilistic framework such as Bayesian networks;* **NMR** (noise and missing value robustness)*: is the algorithm robust w.r.t. noise and missing values;* **QD** (query-driven)*: can the algorithm be used in a query-driven mode;* **DB** (diametrical biclustering)*: indicates if the model explicitly groups genes with both correlated and anticorrelated profiles within the same bicluster.*

5.3 *Pro*Bic model overview

An overview of the *Pro*Bic model is shown in Figure 5.2. The model contains three classes: *Gene*, *Array* and *Expression*. For each class, a set of specific gene, array and expression objects exist that are denoted by lowercase letters g, a and e respectively. The complete set of genes, arrays and expression objects are indicated by uppercase letters G, A and E. Each object g (a) in the *Gene* (*Array*) class, has a number of binary *attributes* B_b that indicate for each gene (array) object if it is part of a bicluster b or not. The *Array* class has an additional attribute *ID* that uniquely identifies each individual array object a. Finally, each object e of the class *Expression* has a single numeric attribute *e.level* that contains the individual expression level value for a specific gene and array. These gene and array objects are specified by two *reference slots* (comparable to *foreign keys* in a database) *e.gene* and *e.array* that point to these specific gene and array objects.

The gene-bicluster $g.B_b$ attributes (over all biclusters b) and the array-bicluster attributes $a.B_b$ are initially unknown and thus hidden variables of the model. The conditional probability distribution $P(e.level|e.gene.B, e.array.B, e.array.ID)$ and the prior distributions $P(\mu_{a,b}, \sigma_{a,b})$, $P(a.B)$ and $P(g.B)$ will be defined in Section 5.4. The model is parameterized with a number of Normal distribution parameters: one set of parameters $(\mu_{a,b}; \sigma_{a,b})$ for each array-bicluster combination (a, b) (see Section 5.4.1).

As described in Section 4.4, for a given gene expression dataset instance, the *Pro*Bic model specifies a joint probability distribution over this particular instance by implicitly defining a ground Bayesian network. Figure 5.3 illustrates how the dataset instance and the *Pro*Bic model implicitly generate a ground Bayesian network (GBN).

The JPD for the *Pro*Bic model is shown in Equation 5.1. For notational convenience, the dependency on the model parameters θ_S is not explicitly written in the CPDs.

$$\begin{aligned} likelihood &= P(E.L, G.B, A.B, A.ID) \\ &= P(E.L|G.B, A.B, A.ID) \cdot P(A.B) \cdot P(G.B) \cdot P(A.ID) \\ &= \prod_{e \in E} P(e.L|e.gene.B, e.array.B, e.array.ID) \cdot \\ &\quad \prod_{a \in A} \prod_{k \in K} P(a.B_k) \cdot \prod_{a \in A} P(a.ID) \cdot \prod_{g \in G} P(g.B) \end{aligned} \tag{5.1}$$

Biclusters in *Pro*Bic are modeled with only a condition-effect and no gene-effect, leading to biclusters with constant columns as defined in Madeira et al. [110]. The biological reason behind this choice is that we are searching for a set of genes with a similar expression pattern across a subset of conditions as these genes are expected to have a common (set of) regulators that are active

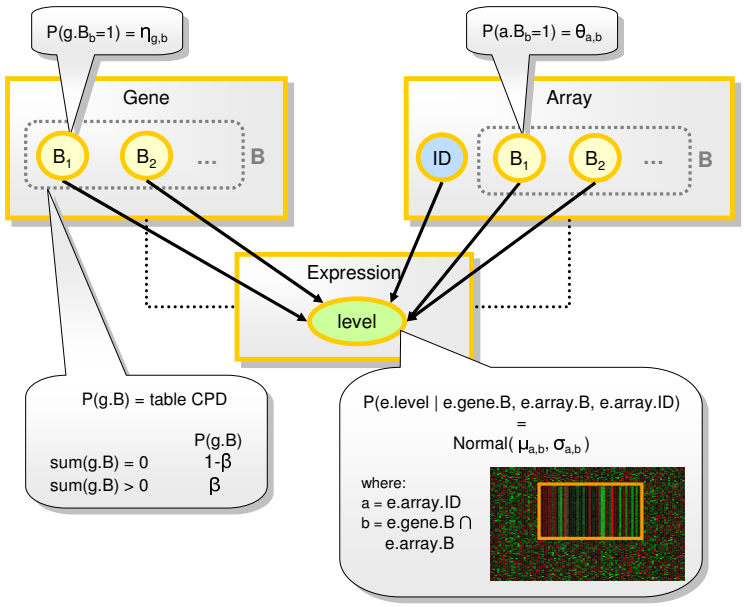


Figure 5.2: ***Schematic overview of the* ProBic *model and the conditional probability distributions of each of the attributes.*** *The three classes of the PRM are Gene, Array and Expression. Expression has one attribute level that contains the expression level and it contains two reference slots gene and array that indicate the gene and the array for which the expression level is measured. Gene and Array each have a number of boolean attributes B_b that represent the presence or absence of respectively a gene or array in a bicluster b. For notational convenience, these attributes are grouped in a vector $B = \{B_1, B_2, ..., B_k\}$. The class Array has an additional attribute ID that uniquely identifies each array. The conditional probability distribution $P(e.level|e.gene.B, e.array.B, e.array.ID)$ is modeled as a set of Normal distributions, one for each array-bicluster combination. A number of prior distributions $P(a.B_b)$, $P(g.B_b)$ and $P(g.B)$ allow expert knowledge to be introduced in the model.*

under that subset of conditions and thereby regulate the genes in the set. The subset of genes and conditions specify a bicluster. As we will see in Section 5.6, the model is easily extended to also incorporate such genes with anticorrelated profiles.

Other biclustering methods such as plaid models [98] also include a gene effect in the model. There are good biological reasons for modeling this gene effect: most microarray data is represented as a log-ratio between a test condition and a reference. Because each gene has a different expression value in the reference condition there is a reference-dependent effect per gene, and one could eliminate this dependency by modeling this gene-effect explicitly. However, gene expression datasets usually have thousands of genes and the addition of a gene effect therefore introduces a huge number of additional parameters

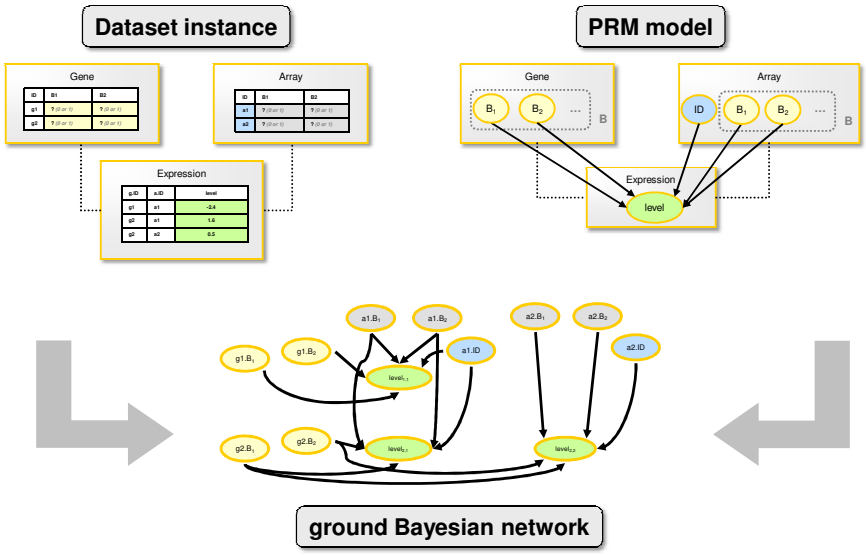


Figure 5.3: ***Schematic overview of the construction of the ground Bayesian network for the* ProBic *model****. In the upper left corner, an example dataset instance is shown with two genes, two arrays and three expression values. Applying the* ProBic *model (upper right corner) to this dataset instance, results in the ground Bayesian network shown in the bottom figure.*

to the model. Due to noise, random correlations will more frequently lead to spurious biclusters in such models with many parameters. Moreover, in microarray compendia that combine multiple series of experiments such as in Lemmens et al. [102], multiple references are used for the different experiment blocks. This would imply an additional layer of parameters where the gene effects are also dependent on the experiment blocks.

In the *Pro*Bic model only condition effects and no gene effects are modeled, thereby avoiding the above problems with over-parametrization of the model. Because of this choice, *Pro*Bic identifies biclusters containing genes that have an expression profile that is differentially expressed *compared to their respective references*. For most microarray experiments, the choice of reference condition reflects a 'good' reference from a biological point of view. For example, heat-shock experiments usually use the undisturbed condition as a reference, experiments where nutrients or chemical compounds are added usually use the nutrient/compound-free condition as reference, etc. The genes with a similar pattern of up- and down-regulation relative to the reference will therefore be biclustered in *Pro*Bic.

5.4 The conditional and prior probability distributions

A detailed overview is given in the following Sections of all probability distributions of the *Pro*Bic model and their associated model parameters.

5.4.1 Expression level CPD P(e.level | e.gene.B, e.array.B, e.array.ID)

This is the main CPD for the biclustering model and it consists of two individual factors:

$$\begin{aligned} &P(e.level|e.gene.B, e.array.B, e.array.ID) = \\ &\quad f_1(e.level|e.gene.B, e.array.B, e.array.ID). \\ &\quad f_2(e.level|e.gene.B, e.array.B, e.array.ID) \end{aligned} \tag{5.2}$$

The first factor $f_1(\ldots)$ describes the main conditional probability of the expression level given the gene- and array-bicluster attributes. The second factor $f_2(\ldots)$ models a penalty factor that reduces the model complexity by decreasing the probability for adding an expression level to a bicluster compared to adding it to the background. In the following sections, each of these factors is explained in detail.

CPD factor 1: $f_1(e.level|e.gene.B, e.array.B, e.array.ID)$

For conveniently defining the CPD $f_1(\ldots)$, three separate cases are identified and a separate definition is given for each of these cases: one for the background distributions, one for bicluster regions without overlap and one for overlapping biclusters. After these three separate definitions, $f_1(\ldots)$ is redefined with a single definition that covers all three cases.

Case 1: background distributions
Let us first evaluate how $f_1(\ldots)$ is modeled in the case where an expression level is part of no bicluster ($e.gene.B \bigcap e.array.B = \emptyset$), i.e. the expression level is part of the *background distributions*. In this case, the probability $f_1(\ldots)$ is modeled as a Normal distribution for each array a, each with a separate set of parameters $(\mu_a^{bgr}, \sigma_a^{bgr})$. The parameters of the background distributions are *fixed* and derived a priori from the dataset using a robust estimation of the background distribution.

For example, two methods can be used for a robust estimation of the background distribution that are based on the *interquartile range normalization* (IQRN) and *smallest quartile range normalization* (SQRN) published by Gu et al. [64]. Both methods assume normally distributed expression values with heavy tails that contain the over- and under-expressed values. These tails are

removed and the mean and variance are estimated based on the truncated distribution which is now assumed to be a truncated Normal distribution. The standard deviation is corrected for removing the $\alpha/2$ tails of the distribution by applying an EM approach that re-estimates the percentage of removed values from the tails *that were part of the background* and the resulting corrected variance until convergence. In IQRN, one first sorts the data in each column, trims off $\alpha/2\%$ of the data from each tail, and computes the trimmed mean and standard deviation. In SQRN, instead of using the middle $(100 - \alpha)\%$ of the data, one first finds for each column the shortest interval that contains a certain percentage (e.g., 50%) of the data. If distributions of the data in each column are symmetric and unimodal, then SQRN is equivalent to IQRN. For skewed distributions, SQRN gives better results [64].

There are a number of reasons why estimating the parameters for the background distribution directly within the model rather than using the robust estimation upfront leads to less optimal results. Initially, the model has not identified any biclusters and therefore uses all expression values of an array as an estimate for the background, including the heavy tails, and thus biases the background estimates towards larger variances. Secondly, some of the EM variants are initialized with the complete dataset as initial bicluster. In this case, no 'background' expression values can be defined as all expression values are part of the initial bicluster. Thirdly, in most practical settings and especially for query-driven biclustering, not all biclusters are identified. The expression values in those unidentified biclusters remain part of the background and therefore contribute to the background estimate.

Case 2: bicluster without overlap
Since there is no overlap between different biclusters in this case, every expression level is part of exactly one bicluster. The probability $f_1(\ldots)$ is again modeled using Normal distributions with parameters (μ, σ) that indirectly depend on the gene- and array-bicluster assignments $(g.B, a.B)$ and on the unique array identifier $a.ID$. So a (μ, σ) value has to be defined for every possible combination of $g.B, a.B, a.ID$.

Since we consider here the case of an expression level that belongs to exactly one bicluster, the combination $g.B, a.B, a.ID$ is uniquely determined by the array a and the bicluster b to which the expression level belongs. As we will see in Section 5.5, this kind of parametrization leads to large computational advantages during the optimizations of the EM algorithm. Formally, the probability for an expression level which is only part of a single bicluster, is

defined as:

$$\begin{aligned} & f_1(e.level|e.gene.B, e.array.B, e.array.ID) \\ = \; & f_1^*(e.level|e.array = a, e.biclusters = \{b\}) \\ = \; & f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}) \\ = \; & \frac{1}{\sigma_{a,b}\sqrt{2\pi}} \exp[-\frac{(e.level - \mu_{a,b})^2}{2\sigma_{a,b}^2}] \end{aligned} \tag{5.3}$$

We introduced the factor $f_1^*(e.level|e.array = a, e.biclusters = \{b\})$ as the probability that an expression level belongs to a single bicluster. The attribute *e.biclusters* does not formally exist in the model, but it is implicitly defined as the set of bicluster indices to which the expression level belongs, namely the intersection $e.array.B \cap e.gene.B$. In this case, the expression level belongs to exactly one bicluster b.

Taking the logarithm of this expression leads to:

$$log f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}) = -log(\sigma_{a,b}) - \frac{(e.level - \mu_{a,b})^2}{2\sigma_{a,b}^2} + constant \tag{5.4}$$

Case 3: overlapping biclusters
The probability $f_1(\ldots)$ in case of overlap can in principle be defined in many different ways. For example, the overlap probability could be modeled as having a high probability for expression values which are close to the sum, the average, the weighted sum, the minimum, the maximum, etc. of the individual bicluster means. It could also be modeled with a distribution that has a separate set of parameters (μ, σ). In this section we will propose a model with some desired properties on both a biological and computational level. Similar to case 2, a probability $f_1(\ldots)$ has to be associated with every possible combination of $g.B, a.B, a.ID$.

Defining a separate set of parameters (μ, σ) for each of the overlap combinations would result in an overfitted model. For example, in case of two overlapping biclusters α and β with a single gene $g_{overlap}$ in the overlap region, a separate parameter set $(\mu_a^{\alpha,\beta}, \sigma_a^{\alpha,\beta})$ would be defined for each condition of the overlap region. Such over-parametrization of the model leads to overfitting and needs to be avoided. A better parametrization would be not to introduce new sets of parameters (μ, σ) for the overlap, but rather to model the overlap region using the parameter sets that were already defined in case 2 (one per array-bicluster combination).

Different overlap models can each be useful from biological perspective and the choice for a particular overlap model depends on the specific research question and the way the gene expression data is normalized. Here we will focus on

the regulatory model as proposed in Chapter 2 and more specifically on the model shown in Figure 2.2. A set consisting of a few genes that are tightly coexpressed in many conditions can be hypothesized to be associated with a highly specific function (second group in Figure 2.2). They consist of genes that respond to the same regulatory program and are coexpressed under many conditions, thereby defining a bicluster with few genes and many conditions. As the bicluster is extended with more genes, the number of conditions under which these genes are coexpressed is expected to decrease. Genes within such extended biclusters only share part of the regulatory program, namely the one that is active under the selected conditions.

In order to model this type of overlap, a slightly more general model is proposed for the overlap than Figure 2.2 depicts. We will define the probability of an expression level in the overlap region here as the geometric mean of the separate bicluster probabilities. In Section 5.5, the mathematical implications of this choice for the EM algorithm are further discussed. Formally, $f_1(\ldots)$ is defined as:

$$\begin{aligned} & f_1(e.level|e.gene.B, e.array.B, e.array.ID) \\ &= \frac{1}{Z_1} \prod_{\substack{b\in \\ \{iset(e.gene.B) \\ \cap \\ iset(e.array.B)\}}} f_1^*(e.level|a,b)^{1/\#\{iset(e.gene.B)\cap iset(e.array.B)\}} \\ &= \frac{1}{Z_1} \prod_{\substack{b\in \\ iset(B_e^i)}} f_1^*(e.level|\mu_{a,b},\sigma_{a,b})^{1/\#iset(B_e^i)} \end{aligned} \tag{5.5}$$

where the following notation was introduced: $iset(X)$, denoting the set of indices i for which the vector elements X_i of binary vector X are 1. B_e^i is defined as the dot product of $e.gene.B$ and $e.array.B$. Therefore, $iset(B_e^i)$ is the set of bicluster-indices in the intersection of $e.gene.B$ and $e.array.B$, or formally: $iset(B_e^i) = iset(e.gene.B) \bigcap iset(e.array.B)$. Finally, $\#iset(B_e^i)$ is the number of elements in this set. Note that the product in Equation 5.5 is only over the biclusters b which the expression level is part of.

Let us now more closely examine the denominator Z which is the normalization function:

$$Z_1 = \int_{-\infty}^{+\infty} \prod_{\substack{b\in \\ iset(B_e^i)}} f_1^*(e.level|\mu_{a,b},\sigma_{a,b})^{1/\#iset(B_e^i)} de \tag{5.6}$$

In the most general case, Z_1 is not constant and depends on the values of the parameters $\mu_{a,b}$ and $\sigma_{a,b}$ that determine the individual bicluster distributions in the overlap. It can however be proven that $Z_1 \simeq 1$ under the assumptions

that the overlap is limited to two biclusters and that the standard deviations of the distributions of the overlapping biclusters are almost identical. In reality, this assumption on the standard deviations only holds approximately and might even be violated in some cases. We will assume in the remainder of this Chapter that this assumption holds and that Z_1 can therefore be considered constant. Other types of overlap could be defined where these assumptions are not necessary.

This type of overlap will assign a high probability to overlapping biclusters where the expression values in the overlap are likely to be in each of the distributions of the individual biclusters. This includes therefore the case where both these distributions are identical (as in Figure 2.2). In cases where the individual bicluster distributions are not identical, the geometric mean of these probabilities will be lower and thus lead to a lower posterior for such models. Nevertheless the model could still identify such overlap cases if the overall probability of the model, including the non-overlapping parts of the bicluster, is sufficiently high.

General case (covering case 1, 2 and 3)
For notational convenience, we will describe a general notation covering all cases 1, 2 and 3:

$$\begin{aligned} & f_1(e.level|e.gene.B, e.array.B, e.array.ID) \\ = & \prod_{\substack{b\in \\ iset(B_e^i)}} f_1^*(e.level|\mu_{a,b}, \sigma_{a,b})^{1/\#iset(B_e^i)} \end{aligned} \tag{5.7}$$

Case 2 is implicitly covered in the notation of case 3 as it can be formulated as a special case of 'overlap' with only one bicluster. For including case 1 in this definition, we introduce a virtual bicluster with index -1 that describes the background as a special kind of bicluster. The parameters $(\mu_{a,b=-1}, \sigma_{a,b=-1})$ of this background bicluster are defined a priori based on the expression data and do not change during optimization. This background bicluster can by definition not overlap with any other biclusters.

The definition of the set $iset(B_e^i)$ also slightly changes as $\prod_{b\in iset(B_e^i)}$ now covers two cases:

- B_e^i empty: background distribution, the product is over the set $b \in [-1]$ so $iset(B_e^i) = [-1]$.
- B_e^i not empty: bicluster distribution, the product is over the set of biclusters in the intersection and never includes $b = -1$ by definition.

CPD factor 2: $f_2(e.level|e.gene.B, e.array.B, e.array.ID)$

The reduction of model complexity in Bayesian networks is usually done by including additional terms in the log-likelihood or log-posterior distributions

such as the Bayesian information criterion (BIC) [142] or the Akaike information criterion (AIC) [3] and these criteria could equally be applied to PRMs. However, the application of these classical techniques for model complexity reduction to the *Pro*Bic model lead to computational intractability if an Expectation-Maximization algorithm is used to find the MAP solution. The reason is that in the substeps of the EM algorithm (Section 5.5), independent optimizations per gene or per condition are not possible anymore if one of the above criteria is included in the model.

Therefore, an alternative strategy is used to reduce model complexity by introducing a 'penalty' factor $f_2(\ldots)$. Without such a penalty factor, the MAP solution for *Pro*Bic would include a very large number of biclusters since each additional bicluster also introduces additional degrees of freedom to the model, thus leading to a higher posterior probability for models with many biclusters. The additional penalty factor $f_2(\ldots)$ is defined such that it only allows a set of expression values to be included in a bicluster if they are on average N times more likely to be in their respective bicluster distributions than in their background distributions. The factor $f_2(\ldots)$ decomposes similarly to $f_1(\ldots)$, leading to the following expression:

$$f_2(e.level|e.gene.B, e.array.B, e.array.ID) = \frac{1}{Z_2} \prod_{b \in iset_B(e)} f_2^*(e.level|b)^{\frac{1}{\#iset_B(e)}} \tag{5.8}$$

where $f_2^*(e.level|b) = \pi_{bgr}$ if the expression level is in the background ($b = -1$) and $f_2^*(e.level|b) = \pi_{bicl}$ if it is in a bicluster ($b \neq -1$). The normalization constant Z_2 is either equal to π_{bgr} (expression level part of background) or to π_{bicl} in all other cases and is thus constant (contrary to Equation 5.6).

This implies that a subset of expression values E_s for a particular array a (or analogously for a gene g), will be assigned to a bicluster q if Equation 5.9 holds:

$$\begin{gathered} \prod_{e \in E_s} f_1^*(e|a, b = q). \prod_{e \in E_s} \pi_{bicl} > \prod_{e \in E_s} f_1^*(e|a, b = -1). \prod_{e \in E_s} \pi_{bgr} \\ \Leftrightarrow \\ \prod_{e \in E_s} \frac{\pi_{bicl}}{\pi_{bgr}} > \prod_{e \in E_s} \frac{f_1^*(e|a,b=-1)}{f_1^*(e|a,b=q)} \end{gathered} \tag{5.9}$$

The user-defined ratio $\frac{\pi_{bicl}}{\pi_{bgr}}$ indicates how many times more likely it must be on average that an expression value is part of the bicluster distribution compared to being part of the background distribution before such a set of expression values E_s is actually added to the bicluster.

Note that using Equations 5.7 and 5.8, we can write Equation 5.10 as:

$$
\begin{aligned}
& P(e.level|e.gene.B, e.array.B, e.array.ID) \\
&= \frac{1}{Z_1 Z_2} \prod_{b \in iset_B(e)} [f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}).f_2^*(e.level|b)]^{\frac{1}{\#iset_B(e)}} && (5.10) \\
&= \frac{1}{Z} \prod_{b \in iset_B(e)} f^*(e.level|\mu_{a,b}, \sigma_{a,b}, b)^{\frac{1}{\#iset_B(e)}} && (5.11) \\
& && (5.12)
\end{aligned}
$$

where $f^*(e.level|\mu_{a,b}, \sigma_{a,b}, b)$ is defined as $f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}).f_2^*(e.level|b)$.

5.4.2 Prior probability for gene to bicluster assignment $P(g.B)$

This prior is also defined as a combination of two factors that each define a separate aspect of the prior. One part of the prior reflects prior knowledge on *specific* gene to bicluster assignments and the other part reflects *general* prior knowledge on gene to bicluster assignments. The prior is defined as follows:

$$P(g.B) = g_1(g.B) \cdot \prod_{b \in B} g_2(g.B_b) \tag{5.13}$$

Specific gene-bicluster assignment prior $g_2(g.B_b)$

We first discuss the *specific* gene to bicluster assignment prior: $g_2(g.B_b = 1)$ is the prior probability for a *particular* gene g to belong to a *particular* bicluster b and it is parameterized as:

$$g_2(g.B_b = 1) = \eta_{g,b} \tag{5.14}$$

This definition implies that for every gene-bicluster combination a specific prior probability can be specified. This prior can for example be used to introduce expert knowledge in the model specifying which genes are highly likely to be in a specific bicluster. In cases where a researcher expects a set of genes to be coregulated, e.g. based on a common set of motifs or based on ChIP-chip binding experiments, these genes can be preferentially assigned to a specific bicluster b by increasing the $\eta_{g,b}$ values for each coregulated gene g. The opposite case could also occur where a researcher knows/expects that some genes should not be part of a particular bicluster. By decreasing the $\eta_{g,b}$ values for those genes g, this prior knowledge can be expressed in the model.

While this prior could theoretically also be used for *query-driven biclustering*, there are a number of practical reasons why such prior is not well suited. The prior's effect on the JPD is highly 'local' and creates peaks of high probability

in the posterior distribution that are 'hidden' in adjacent[2] low probability regions of the posterior. Any approximate method that iteratively samples points in this distribution to find the MAP solution and where subsequent points are 'near' to one another in the posterior landscape, tend to be trapped in other local optima with a far lower posterior than the global optimum. In Section 5.4.5, a set of distribution priors will be introduced that do not suffer from this problem and are therefore more suitable for query-driven biclustering as will be discussed in Section 5.5.4.

General gene-bicluster assignment prior $g_1(g.B)$

The other factor $g_1(g.B)$ is the prior probability that *a* gene is part of *a* bicluster. This prior indirectly has an effect on the average number of genes in a bicluster. The motivation for this prior is biological in nature: biologists are often interested in information concerning specific pathways, typically containing between 5 and 100 genes. By penalizing the addition of genes to a bicluster, the average number of genes in a bicluster can be reduced, keeping only the best fitting genes in the bicluster profile. $g_1(g.B)$ is parameterized in the following way:

$$g_1(g.B) = \begin{cases} 1-\beta, & \text{if } sum(g.B) = 0 \\ \beta, & \text{if } sum(g.B) \neq 0 \end{cases} \tag{5.15}$$

where a value $\beta < 0.5$ introduces a penalty for a gene belonging to one or more biclusters.

One might expect that this type of penalization could also be achieved using the $g_2(g.B_b)$ prior distributions, by setting $g_2(g.B_b = 1)$ to a lower probability for all genes g than $g_2(g.B_b = 0)$. This kind of use of the $g_2(g.B_b)$ prior has however an undesired side-effect. When a gene is assigned to more than one bicluster (N biclusters), the prior probability $g_2(g.B_b = 1)$ is counted N times in Equation 5.1. For large penalty values, meaning low $P(g.B_b = 1)$ probabilities, assigning genes to more than one bicluster becomes highly unlikely as a result.

5.4.3 Prior probability for array to bicluster assignment $P(a.B_b)$

Similar to the prior on gene to bicluster assignments in Section 5.4.2, $P(a.B_b)$ is the prior probability for a *specific* array *a* to belong to a specific bicluster *b*:

$$P(a.B_b = 1) = \begin{cases} \zeta, & \text{if } a.B_b = \mathbf{0} \\ 1-\zeta, & \text{else} \end{cases} \tag{5.16}$$

While this prior has not been explicitly used in any of the presented results in Section 5.7, it could in principle be used in a similar way as the $g_1(g.B)$

[2] 'Adjacent' is defined with respect to having small differences in $g.B$ or $a.B$ assignments. Any algorithm that walks through the solution space jumps from a point to one of these adjacent points during optimization.

prior, namely to increase or reduce the average number of conditions in the biclusters. However, in most practical settings the ratio $\frac{\pi_{bicl}}{\pi_{bgr}}$ is the parameter that has most effect on the number of arrays in a bicluster.

5.4.4 Prior for the array identifiers $P(a.ID)$

Each array is given a unique identifier $a.ID$ and in principle a prior distribution can be defined over these arrays. However, the use of such prior is very limited and for the *Pro*Bic model this prior distribution is therefore chosen uniform. As a consequence this prior does not contribute to any of the optimization steps and is also not explicitly written in the equations.

5.4.5 Prior for the model parameters $P(\theta)$

The *Pro*Bic model is parameterized by a set of parameters $(\bar{\bar{\mu}}, \bar{\bar{\sigma}})$, one set (μ_{ab}, σ_{ab}) per array a and per bicluster b. Biological and expert knowledge can be introduced in the model through proper prior distributions. As we will see in Section 5.7, these prior distributions are well-suited for query-driven biclustering.

A straightforward decomposition of the prior is to assume a similar structure as the expression level CPD, leading to Equation 5.17. Similar to Equation 5.7, the product $\prod_{b \in B}$ ranges over all the biclusters including the background distribution. Based on this decomposition, the individual distributions $P(\mu_{a,b}, \sigma_{a,b})$ can now be chosen such that they are conjugate to the expression level CPD of Equation 5.7.

$$\begin{aligned} P(\theta) &= P(\bar{\bar{\mu}}, \bar{\bar{\sigma}}) \\ &= \prod_{a \in A} \prod_{b \in B} P(\mu_{a,b}, \sigma_{a,b}) \end{aligned} \tag{5.17}$$

Any member of the exponential family is a conjugate prior distribution to Equation 5.17, e.g. the Normal, scaled inverse χ^2, Gamma, Inverse Gamma and Normal-Gamma distributions are all conjugate to the Normal distribution. In Appendix A.3, the main properties of some interesting priors are listed.

While the interpretation is slightly more difficult than is the case for a Normal distribution, we will highlight the use of a Normal-Inverse-χ^2 distribution here as it is a particularly useful type of prior for the biological problems of interest. It defines a prior distribution directly for the mean and standard deviation and is parameterized by the set of hyperparameters $(\mu_0, \kappa_0, \nu_0, \sigma_0^2)$. The mean and standard deviation are distributed in the following way:

$$\sigma^2 \qquad Inv - \chi^2(\nu_0, \sigma_0^2) \tag{5.18}$$

$$\mu|\sigma^2 \qquad N(\mu_0, \sigma^2/\kappa_0) \tag{5.19}$$

Using this prior distribution, a researcher can for example perform *query-driven biclustering* by requiring the means of the distributions for a particular bicluster to be centered around the means of the query genes (see Section 5.5.4 for more details). Secondly, the prior can also be used to select biclusters with distributions that are more tightly co-expressed than the background distributions by specifying a prior on the variance through σ_0 and ν_0 (see Appendix A.3). For example, the results for query-driven biclustering in Section 5.7.5 apply this prior distribution with the default parameter settings for query-driven biclustering as specified in Section 5.5.4.

5.4.6 Posterior distribution $P(M|D)$

The following log(posterior) distribution is obtained by combining the prior (Equation 5.17) and the likelihood function (Equation 5.1):

$$\begin{aligned} log(posterior) \quad = \quad & \sum_{a \in A} \sum_{b \in B} logP(\mu_{a,b}, \sigma_{a,b}) + \\ & \sum_{e \in E} \sum_{b \in B_e^i} \frac{log[f^*(e.level|\mu_{a,b}, \sigma_{a,b})]}{\#iset(B_e^i)} + \\ & \sum_{b \in B} \sum_{a \in A} logP(a.B_b) + \sum_{a \in A} logP(a.ID) + \\ & \sum_{b \in B} \sum_{g \in G} log[g_2(g.B_b)] + \sum_{g \in G} log[g_1(g.B)] + C \end{aligned} \tag{5.20}$$

where C is a constant.

The hidden variables in Equation 5.20 are the gene to bicluster assignments $g.B_b$ and the array to bicluster assignments $a.B_b$. The model parameters are the Gaussian distribution parameters $(\bar{\mu}, \bar{\sigma})$, with one set of $(\mu_{a,b}, \sigma_{a,b})$ values per array a and bicluster b.

The calculation of the full posterior distribution over all possible assignments of hidden variables is exponential in both the number of genes and arrays and thus computationally intractable. However, the computational cost can be drastically reduced by only deriving the MAP solution of this function w.r.t. the model parameters and the hidden variables. For the *Pro*Bic model, we will outline an Expectation-Maximization strategy in Section 5.5 and a detailed discussion will be given about the advantages and some disadvantages of using EM for the *Pro*Bic model and for PRMs in general.

5.5 Learning the model: Expectation-Maximization (EM) algorithm

An attractive approach that is specifically tailored to optimizing likelihood functions in case of missing values, is Expectation-Maximization. It deals with missing or hidden values by replacing them by an estimated value or by an estimated distribution over all values given the current model parameters and then it recalculates the model parameters given these estimated values. This process is repeated until the values converge. The EM procedure is guaranteed to converge to a local optimum under fairly general conditions [43, 171].

We will apply a *hard assignment* Expectation-Maximization (EM) approach. Hard assignment means that a single value is assigned to the hidden variables during the expectation step rather than a distribution over all possible values as is the case in *soft assignment* EM.

Within the PRM framework, EM has some additional advantages. Firstly, due to the design of the model, the log-posterior decomposes into a number of independent terms either per gene or per array in every substep of the EM algorithm. The independent optimization of these terms greatly reduces the computational cost. Secondly, the model was designed such that when extending the model with additional data sources, most of the EM steps remain unchanged if some model design constraints are met (see Section 5.8). Thirdly, while EM only guarantees a local optimum of the posterior, our results on artificial data indicate that under fairly general conditions the global optimum or near-optimum is usually found in practice.

Other methods such as Gibbs sampling have also been successfully applied for biclustering gene expression data [47, 150]. While Gibbs sampling is less sensitive to local optima and will generally lead to better solutions, it is also typically slower than an Expectation-Maximization approach.

The EM iterates the following steps until convergence:

- Maximization step: maximize over the model parameters $\bar{\mu}, \bar{\sigma}$, keeping the hidden variables (i.e. the gene- and array-bicluster assignments $G.B$ and $A.B$) fixed.

- Expectation step: find the expected values for the hidden variables $G.B$ and $A.B$, keeping the current model parameters $\bar{\mu}, \bar{\sigma}$ fixed.

5.5.1 Maximization step

In the maximization step, all hidden variables $A.B$ and $G.B$ are assumed to be known either from the initialization or from a previous Expectation step. We need to maximize the log-posterior (Equation 5.20) w.r.t. the parameters $\bar{\mu}$ and

$\bar{\bar{\sigma}}$ for the given $G.B$ and $A.B$ assignment:

$$
\begin{aligned}
& argmax_{\bar{\mu},\bar{\bar{\sigma}}} log(posterior) \\
= \; & argmax_{\bar{\mu},\bar{\bar{\sigma}}} \\
& \sum_{a \in A} \sum_{b \in B} logP(\mu_{a,b}, \sigma_{a,b}) + \sum_{b \in B} \sum_{a \in A} logP(a.B_b) + \\
& \sum_{e \in E} \sum_{b \in B_e^i} \frac{log f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}).f_2^*(e.level|b)}{\#iset(B_e^i)} + \\
& \sum_{g \in G} logg_1(g.B) + \sum_{b \in B} \sum_{g \in G} logg_2(g.B_b)
\end{aligned} \tag{5.21}
$$

Equation 5.21 can be simplified through the following observations. Several terms are constant for fixed $A.B$ and $G.B$ assignments and due to our particular choice of overlap model, one pair of $(\mu_{a,b}, \sigma_{a,b})$ values is defined per array-bicluster combination. From these observations, the following optimization problem is derived from Equation 5.21:

$$
\begin{aligned}
& argmax_{\bar{\mu},\bar{\bar{\sigma}}} log(posterior) \\
= \; & argmax_{\bar{\mu},\bar{\bar{\sigma}}} \\
& \sum_{a \in A} \sum_{b \in B} logP(\mu_{a,b}, \sigma_{a,b}) + \\
& \sum_{a \in A} \sum_{e \in E_{\cdot a}} \sum_{\substack{b \\ \in \\ iset(B_e^i)}} \frac{log f_1^*(e.level|\mu_{a,b}, \sigma_{a,b})}{\#iset(B_e^i)}
\end{aligned} \tag{5.22}
$$

where $E_{\cdot a}$ is defined as the subset of expression values for array a.

Equation 5.22 can be maximized independently per array since the parameters $(\mu_{a,b}, \sigma_{a,b})$ are independent of each other, leading to the following optimization problem *per array*:

$$
argmax_{\bar{\mu}_a,\bar{\sigma}_a} \sum_{b \in B} logP(\mu_{a,b}, \sigma_{a,b}) + \sum_{e \in E_{\cdot a}} \sum_{\substack{b \\ \in \\ iset(B_e^i)}} \frac{log f_1^*(e.level|\mu_{a,b}, \sigma_{a,b})}{\#iset(B_e^i)} \tag{5.23}
$$

The second summation in the above expression can be written independently per bicluster by applying the following procedure. The expression levels are grouped per different B_e^i assignment *for the particular array which is optimized*. This means: group over all different possible $B_e^i (= e.gene.B \cap e.array.B)$ sets (for a given array a, over all genes g). An example is given in Figure 5.4 and illustrates the different B_e^i groups for two choices of array a.

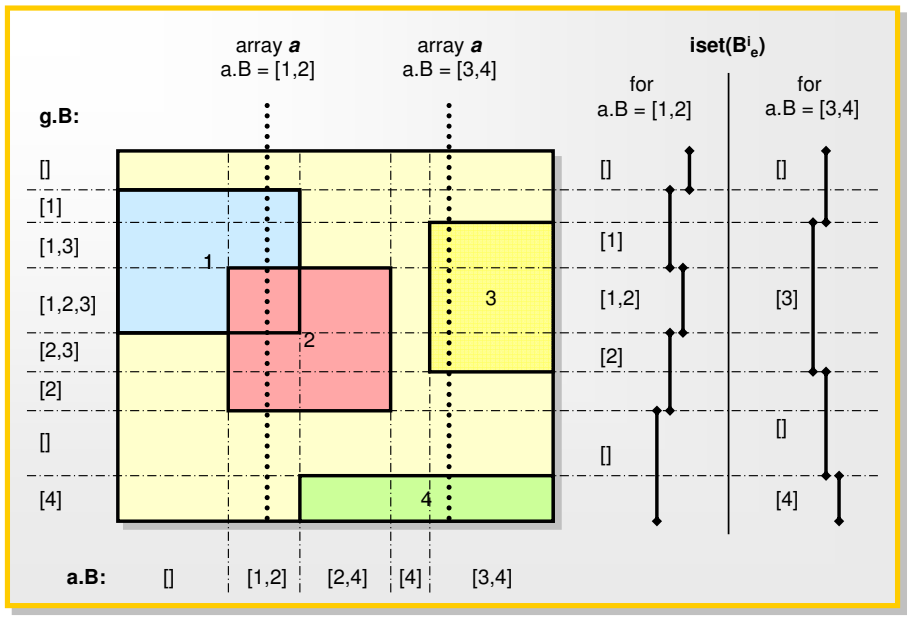


Figure 5.4: Illustration of the expression level grouping in the optimization of the Maximization step. For reducing notational overhead, the binary vectors a.B and g.B are represented here as the set of indices for which the vector elements are equal to 1. The image represents a gene by array matrix where the arrays and genes with different a.B and g.B sets respectively, are indicated. For two specific arrays, iset(B_e^i) is shown for each of the g.B possibilities.

Observing Equation 5.23, we note that there is a sum over one or more biclusters for every possible B_e^i set. The summations are rearranged accordingly and lead to the following optimization problem per array:

$$argmax_{\bar{\mu}_a,\bar{\sigma}_a} \sum_{b\in B} logP(\mu_{a,b},\sigma_{a,b}) + \sum_{b\in B} \sum_{\substack{e\in E_{\cdot a}:\\ b\in iset(B_e^i)}} \frac{log f_1^*(e.level|\mu_{a,b},\sigma_{a,b})}{\#iset(B_e^i)} \tag{5.24}$$

Because of this rearrangement and because the parameters $(\mu_{a,b},\sigma_{a,b})$ are independent per bicluster, Equation 5.24 can now be optimized independently per bicluster. The final set of optimization problems is given by Equation 5.25, one per array-bicluster combination, and is generally applicable for any type

of prior as defined in Section 5.4.5.

$$argmax_{\mu_{a,b},\sigma_{a,b}} logP(\mu_{a,b},\sigma_{a,b}) + \sum_{\substack{e\in E_{\cdot a}:\\ b\in iset(B_e^i)}} \frac{log f_1^*(e.level|\mu_{a,b},\sigma_{a,b})}{\#iset(B_e^i)} =$$

$$argmax_{\mu_{a,b},\sigma_{a,b}} logP(\mu_{a,b},\sigma_{a,b}) - n_w log(\sigma_{a,b}) - \frac{ssq_w - 2\mu s_w + n_w\mu_{a,b}^2}{2\sigma_{a,b}^2} \tag{5.25}$$

where $E_{\cdot a}$ is defined as the subset of expression values for array a and where the following variables were used:

$$\text{(weighted) count: } n_w = \sum_{\substack{e\in E_{\cdot a}:\\ b\in iset(B_e^i)}} \frac{1}{\#iset(B_e^i)} \tag{5.26}$$

$$\text{(weighted) sum: } s_w = \sum_{\substack{e\in E_{\cdot a}:\\ b\in iset(B_e^i)}} \frac{e.level}{\#iset(B_e^i)} \tag{5.27}$$

$$\text{(weighted) sum of squares: } ssq_w = \sum_{\substack{e\in E_{\cdot a}:\\ b\in iset(B_e^i)}} \frac{(e.level)^2}{\#iset(B_e^i)} \tag{5.28}$$

We will now derive the analytical expressions for the case where the prior distribution $P(\bar{\mu},\bar{\sigma})$ is a Normal distribution, characterized by its sufficient statistics: count n_0, sum s_0 and sum of squares ssq_0. For this choice of prior, the optimization problem per array and per bicluster becomes:

$$argmax_{\mu_{a,b},\sigma_{a,b}} - (n_w + n_0)log(\sigma_{a,b}) - \frac{(ssq_w + ssq_0) - 2\mu(s_w + s_0) + (n_w + n_0)\mu_{a,b}^2}{2\sigma_{a,b}^2} \tag{5.29}$$

Taking the derivative of the above expression w.r.t. $\mu_{a,b}$ and $\sigma_{a,b}$, leads to the following set of analytical solutions for each $(\mu_{a,b},\sigma_{a,b})$ pair:

$$\mu_{a,b} = \frac{s_w + s_0}{n_w + n_0} \tag{5.30}$$

$$\sigma_{a,b}^2 = \frac{(ssq_w + ssq_0) - \frac{(s_w+s_0)^2}{n_w+n_0}}{n_w + n_0} \tag{5.31}$$

5.5.2 Expectation step

The Expectation step needs to maximize the posterior w.r.t. all possible assignments for the $G.B$ and $A.B$ vectors. This optimization problem is computationally intractable. However, a generalized Expectation-Maximization

(GEM) approach [25, 113] can be applied that splits the Expectation-step into two substeps. In these substeps the gene-bicluster attributes $G.B$ and the array-bicluster attributes $A.B$ are alternatively fixed and the posterior is maximized w.r.t. the other attributes ($A.B$ and $G.B$ respectively). This approach leads to a further decomposition of the log-posterior in each of the substeps into a number of terms that can independently optimized:

- E step 1: maximize posterior w.r.t. array-bicluster attributes $A.B$, keeping $\bar{\mu}, \bar{\sigma}$ and $G.B$ fixed.
- E step 2: maximize posterior w.r.t. gene-bicluster attributes $G.B$, keeping $\bar{\mu}, \bar{\sigma}$ and $A.B$ fixed.

E-step 1: reassigning the $A.B$ attributes

In this step, the model parameters ($\bar{\mu}, \bar{\sigma}$) and the gene-bicluster attributes $G.B$ are fixed. We need to maximize the posterior w.r.t. all possible $A.B$ assignments, given the current $G.B$ attributes and model parameters:

$$
\begin{aligned}
& argmax_{A.B} log(posterior) \\
= \; & argmax_{A.B} \\
& \sum_{a \in A} \sum_{b \in B} logP(\mu_{a,b}, \sigma_{a,b}) + \sum_{e \in E} \sum_{b \in B_e^i} \frac{logf^*(e.level | \mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \\
& \sum_{b \in B} \sum_{a \in A} logP(a.B_b) + \sum_{b \in B} \sum_{g \in G} logg_2(g.B_b) + \sum_{g \in G} logg_1(g.B) \\
= \; & argmax_{A.B} \\
& \sum_{a \in A} \sum_{e \in E_a} \sum_{b \in B_e^i} \frac{logf^*(e.level | \mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \sum_{a \in A} \sum_{b \in B} logP(a.B_b) \qquad (5.32)
\end{aligned}
$$

The above expression can be optimized independently per array:

$$
argmax_{a.B} \sum_{e \in E_a} \sum_{b \in iset(B_e^i)} \frac{logf^*(e.level | \mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \sum_{b \in B} logP(a.B_b) \qquad (5.33)
$$

Equation 5.33 needs to be evaluated for every possible $a.B$ assignment. This would lead to $2^{\#B}$ evaluations if there are $\#B$ biclusters. Note that in Equation 5.33, B_e^i also indirectly depends on the $a.B$ assignment, since it is the intersection between $iset(a.B)$ and $iset(g.B)$.

Optimization 1: precalculating and caching intermediate values

The most demanding calculations in Equation 5.33 can be performed once upfront, thereby greatly reducing the computational cost. Equation 5.33 can be rewritten by grouping terms with the same gene-bicluster assignment $g.B$ together: the set of bicluster indices $iset(B_e^i)$ for an expression level e is identical

for all genes in such a group, independent of the choice of $a.B$. Note that the set $iset(B_e^i)$ does change with a different choice of $a.B$, but it changes in the same way for all genes that have an identical $g.B$ vector, so for a given $g.B$, $iset(B_e^i)$ only depends on $a.B$. This grouping allows the reordering of summations and leads to the definition of some invariant sums over the different $a.B$ evaluations:

$$\begin{aligned}
& argmax_{a.B} \sum_{\substack{gb \in \\ unq(G.B)}} \sum_{\substack{e \in E_a: \\ e.gene \in gb}} \sum_{b \in iset(B_e^i)} \frac{logf^*(e.level|\mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \sum_{b \in B} logP(a.B_b) \\
= \; & argmax_{a.B} \sum_{\substack{gb \in \\ unq(G.B)}} \sum_{b \in iset(B_e^i)} \frac{1}{\#iset(B_e^i)} \sum_{\substack{e \in E_a: \\ e.gene \in gb}} logf^*(e.level|\mu_{a,b}, \sigma_{a,b}, b) + \sum_{b \in B} logP(a.B_b) \\
= \; & argmax_{a.B} \sum_{\substack{gb \in \\ unq(G.B)}} \sum_{b \in iset(B_e^i)} \frac{v_{a,gb,b}}{\#iset(B_e^i)} + \sum_{b \in B} logP(a.B_b) \\
= \; & argmax_{a.B} \sum_{\substack{gb \in \\ unq(G.B)}} \sum_{b \in iset(B_e^i)} \frac{v_{a,gb,b}}{\#iset(B_e^i)} + N_a^0 \pi_{bgr} + (\#B - N_{a,b}^0)(1 - \pi_{bgr}) \qquad (5.34)
\end{aligned}$$

where each of the values $v_{a,gb,b}$ is precalculated and N_a^0 is the number of biclusters b for which $a.B_b = 0$.

Figure 5.5 illustrates this precalculation step for an example with two biclusters. The summations for a particular $a.B$ in Equation 5.34 are decomposed into a weighted sum of precalculated values $v_{a,gb,b}$, which are represented by the colored blocks on the right hand side for a particular array a. For each $a.B$ vector, the expression of Equation 5.34 is visually represented as a column with the weighted contributions of each $v_{a,gb,b}$. For example, the resulting expression for $a.B = [1, 2]$ is:

$$v_{a,[1],[1]} + \frac{v_{a,[1,2],[1]} + v_{a,[1,2],[2]}}{2} + v_{a,[2],[2]} + v_{a,[],[]} \qquad (5.35)$$

Optimization 2: Apriori algorithm
While the first optimization reduces the computational cost per $a.B$ evaluation, a second optimization can reduce the amount of evaluations by orders of magnitude. Intuitively, one could hypothesize that assigning an array to two biclusters X and Y will not lead to an improvement in score if the score of assigning the array to both a single bicluster X and a single bicluster Y does not lead to an improvement respectively (compared to the background assignment).

One can prove that the above hypothesis holds under fairly general conditions when the overlap between biclusters remains relatively small compared to the non-overlapping parts of the biclusters. See Appendix A.4 for a formal definition of this hypothesis and the derivation of the necessary conditions for

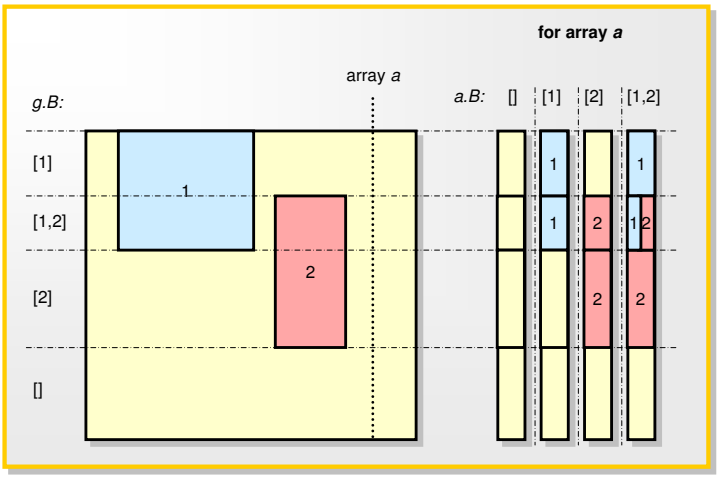


Figure 5.5: ***Illustration of the E-step 1.*** *For a particular array a in the example, we need to evaluate Equation 5.33 for all possible a.B assignments. At the right hand side the situation is shown for array a for each of the possible a.B assignments, namely* $[], [1], [2]$ *and* $[1,2]$*. Equation 5.33 consists of contributions from each of the expression levels in the colored blocks. Blocks of identical color represent the same contributions for each of the a.B assignments in Equation 5.34.*

this hypothesis. An *Apriori*-like procedure, as described below, results in the maximum a posteriori solution, namely if the overlap between two biclusters is small compared to the respective bicluster sizes.

An approach similar to the *Apriori* algorithm [2] is used to evaluate only a small subset of $a.B$ vectors with potentially good scores while still guaranteeing the optimality of the solution. For notational convenience, the binary vector $a.B$ is here represented as a vector of bicluster indices for which the vector elements are equal to 1, e.g. a vector $a.B = (1,0,0,1,0)$ is represented as $[1,4]$. The following procedure is applied: in the first iteration (level 0), the score for the background assignment $a.B = []$ is calculated. For all next levels, the following steps are iterated until no more candidate assignments remain and the $a.B$ assignment with the highest score is selected as the maximum a posteriori solution of Equation 5.34.

- At level L, a list of candidate vectors $a.B$ with each L bicluster indices is constructed from the set of retained vectors from the previous level $(L-1)$ using the following procedure. A vector $a.B$ is added to the list of candidate vectors if each of the L subvectors of $a.B$ with $(L-1)$ elements

(containing all but one of the indices of $a.B$) is present in the retained vectors from the previous level.

- The score is calculated for each of these candidate vectors according to Equation 5.34 and the candidate vector $a.B$ is retained for the next level only if its score is higher than the score of each of the $(L-1)$ subvectors. An $(L-1)$ subvector of $a.B$ contains $(L-1)$ biclusters that are all a bicluster of $a.B$ and is part of the retained vectors from level $(L-1)$.
- The list of retained vectors is used to construct the candidate vectors of the next level $(L+1)$.

E-step 2: reassigning the $G.B$ attributes

In this step, the model parameters $(\bar{\mu}, \bar{\sigma})$ and all the array-bicluster attributes $A.B$ are fixed. We now need to maximize the posterior w.r.t. all possible $G.B$ assignments:

$$argmax_{G.B} \sum_{g \in G} \sum_{e \in E_{g\cdot}} \sum_{b \in B_e^i} \frac{logf^*(e.level|\mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \sum_{g \in G} \sum_{b \in B} logg_2(g.B_b) + \sum_{g \in G} logg_1(g.B) \tag{5.36}$$

where $E_{g\cdot}$ is defined as the subset of expression values for gene g.

This expression can be optimized independently per gene and similar optimizations as in Section 5.5.2 are performed:

$$\begin{aligned} & argmax_{g.B} \sum_{\substack{ab \in \\ unq(A.B)}} \sum_{\substack{e \in E_{g\cdot}: \\ e.array \in ab}} \sum_{b \in B_e^i} \frac{logf^*(e.level|\mu_{a,b}, \sigma_{a,b}, b)}{\#iset(B_e^i)} + \\ & \sum_{b \in B} logg_2(g.B_b) + logg_1(g.B) \\ = \; & argmax_{g.B} \sum_{\substack{ab \in \\ unq(A.B)}} \sum_{b \in B_e^i} \frac{1}{\#iset(B_e^i)} \sum_{\substack{e \in E_{g\cdot}: \\ e.array \in ab}} logf^*(e.level|\mu_{a,b}, \sigma_{a,b}, b) + \\ & \sum_{b \in B} logg_2(g.B_b) + logg_1(g.B) \\ = \; & argmax_{g.B} \sum_{\substack{ab \in \\ unq(A.B)}} \sum_{b \in B_e^i} \frac{w_{g,ab,b}}{\#iset(B_e^i)} + \sum_{b \in B} logg_1(g.B_b) + \beta^\kappa (1-\beta)^{(1-\kappa)} \end{aligned} \tag{5.37}$$

where κ is defined as 1 if at least one element differs from 0 in the vector $g.B$, otherwise it is 0. $w_{g,ab,b}$ is defined analogously to $v_{a,gb,b}$ in Section 5.5.2 and it is

also precalculated to reduce the computational cost. A similar Apriori strategy as in Section 5.5.2 is applied to greatly reduce the number of evaluations of Equation 5.37.

5.5.3 EM initialization

Any Expectation-Maximization algorithm starts the optimization procedure from an initial assignment of the hidden variables. For the *Pro*Bic model, these are the gene- and condition-bicluster attributes that require an initialization. Once these attributes are known, the algorithm starts with a Maximization step in which the parameter matrices $\bar{\mu}, \bar{\sigma}$ are estimated. Several strategies can be applied to initialize the bicluster attributes $G.B$ and $A.B$:

- Random initializations.
- Initialization around known gene clusters, known biclusters or around a set of seed genes.
- Initialization with the complete set of genes and arrays.

Depending on the model and the specific dataset that is used, the type of initialization could affect the biclustering result. Ideally, the initialization of the algorithm is chosen such that the optimization result is independent of the initialization. Results on simulated data have shown that for global biclustering, the initialization of each bicluster preferably contains the complete set of genes and conditions as this converges to the global optimum in most practical settings and is on average more likely to converge to the global optimum than a set of random initializations (results not shown).

5.5.4 Query-driven biclustering in *Pro*Bic

The *Pro*Bic model can be used to perform *query-driven biclustering* by using the prior distributions as defined in Section 5.4.5. Starting from a set of *query genes* that are of interest to a biologist, the model searches for a bicluster with a similar expression profile as the query genes for a subset of arrays. The distribution prior $P(\bar{\mu}, \bar{\sigma})$ can be any member of the exponential family of which two priors are particularly interesting: the Normal distribution and the Normal-Inverse-χ^2 distribution (see Appendix A.3).

The Normal distribution is the most intuitive to be used as prior, since it can be considered as a prior that adds a number of pseudo expression values to the data. It is parameterized by a mean μ_0, a standard deviation σ_0 and a count n_0 where the count represents the number of pseudo values. The Normal-Inverse-χ^2 distribution is more complex but it gives the researcher more freedom to e.g. direct the algorithm towards biclusters with tighter expression profiles. It is parameterized by the hyperparameters $(\mu_0, \kappa_0, \nu_0, \sigma_0^2)$. μ_0 reflects the prior

mean and σ_0^2/κ_0 reflects the a priori variance on this mean. The parameters ν_0 and σ_0 determine the a priori variance of the distribution and its associated variance. See Appendix A.3 for more details.

The Normal-Inverse-χ^2 distribution is used for all results presented in this thesis. By choosing the prior mean $\mu_{a,b}^0$ as the sample average μ_a^{query} for the expression values defined by the set of query genes for each array a, the algorithm will identify a bicluster b around the expression profile of these query genes. The prior standard deviation σ_a^{query} is by default chosen to be smaller than the background standard deviation σ_a^{bgr} by a fraction f_{bcl} (set to 0.4 for all presented results) in order to identify tight bicluster profiles. The user can however define priors that select genes that are more stringently or more loosely tied around the expression profile of the query genes by varying σ_a^{query}.

From a biological perspective, this definition of query-driven biclustering has an interesting property: some query genes can be removed from the bicluster if they do not fit the bicluster profile well. This is very useful in situations where a biologist is interested in identifying all the genes that are involved in a specific pathway and where a known set of query genes is available that are *expected* to be part of that pathway, meaning that most of these query genes are indeed in the pathway but some are outliers that do not belong to the pathway. The application of such 'noisy' queries with outlier genes on the *E. coli* compendium will be given in Section 5.7.

5.5.5 Convergence speed and quality of local optimum

The convergence of the EM algorithm can be adversely affected by 'disruptive' changes during the EM iterations. For example, the complete removal of all genes or arrays from a bicluster in one of the expectation substeps is such a disruptive change. Large changes in the gene-bicluster assignments during the Expectation steps can cause large changes in the $(\bar{\mu}, \bar{\sigma})$ parameter values during the Maximization step. Large changes in the array-bicluster assignments do not affect the $(\bar{\mu}, \bar{\sigma})$ parameter values, but indirectly also affect the convergence behavior as this can lead to a different gene-bicluster assignment in E-step 2. Analogous to large temperature changes in simulated annealing, large variations in both the model parameters $(\bar{\mu}, \bar{\sigma})$ and the hidden variables $G.B$ and $A.B$ can lead to a local optimum with a lower scores compared to an approach with smaller variations.

In order to avoid such disruptive changes, only a limited number of genes-bicluster or array-bicluster attributes are allowed to be changed during the Expectation steps. If we more closely examine the log-posterior formula of the *Pro*Bic model (Equation 5.20), it is not the overall change in gene-bicluster or array-bicluster assignments that needs to be limited, but rather the change in assignments *per* gene or array *block* respectively where a *block* of genes (or

arrays) is defined as a set of genes (or arrays) with the same gene- (or array-) bicluster assignment.

Two user-defined parameters in *Pro*Bic control the maximally allowed percentage of genes or arrays that can be changed during the Expectation steps. If more genes or arrays are to be changed in one of these steps, only the N top genes or arrays that lead to the largest increase in score are allowed to change their bicluster assignment.

5.5.6 EM algorithm variants for query-driven and global biclustering

Several variants of the EM algorithm (Figure 5.6) were evaluated with respect to the quality of the identified local optimum and the convergence stability of the algorithm for both global biclustering and query-driven biclustering settings. Various experiments on simulated data have lead to the following conclusions.

Variant A:	**Variant B:**	**Variant C:**
initialization	*initialization*	*initialization*
while(not converged):	*while(not converged):*	*while(not converged):*
maximization	*maximization*	*maximization*
expectation-AB	*expectation-AB*	*expectation-GB*
maximization	*while(not converged):*	*while(not converged):*
expectation-GB	*maximization*	*maximization*
	expectation-AB	*expectation-AB*
	maximization	*maximization*
	expectation-GB	*expectation-GB*

Figure 5.6: ***Different variants of the EM algorithm.***

For global biclustering, the best results are obtained using *Variant C*, in which first only the gene-bicluster attributes are optimized and in the second phase both gene- and array-bicluster attributes are updated simultaneously in the optimization procedure. These results match with what one would expect from a theoretical perspective. At each iteration i during the first phase, the current bicluster consists of a subset of genes over all conditions. Now consider an array that is not part of the real bicluster. Assuming the prior distributions for background and biclusters are either not too strong or sufficiently similar to one another, the model scores for a model M_1 with this array and for a model M_2 without this array, will not differ much. The reason is that for arrays that are not part of the bicluster, the parameters of these 'bicluster' distributions are very similar to those the background. Their differences are only due to statistical variations of the (sampled) bicluster expression values for that array. Therefore, the *in*clusion of *background* arrays in the bicluster will have little effect on the optimization algorithm. However, the *ex*clusion

of *bicluster* arrays may lead too poor convergence. Therefore *Variant C* has far better convergence properties than the two other variants in Figure 5.6.

For query-driven biclustering, *Variant B* achieves the best results. The algorithm starts with a set of query genes and all conditions as the initial bicluster. In the first phase, only condition-bicluster attributes are changed and this results in the removal of conditions whose expression levels are not sufficiently different enough from the background distribution. Note that 'sufficiently different' is determined by the ratio $\frac{\pi_{bicl}}{\pi_{bgr}}$ as defined in Equation 5.9.

One might expect that the convergence properties of *Variant C* would also be good for query driven biclustering, however in some cases the necessary conditions as described in the previous paragraph are not met, namely that for these query genes the bicluster distributions are similar to the background for all arrays that are not part of the real bicluster. The reason is that the set of query genes is usually very small, typically 1-5 genes, and thus statistical variations on the expression values have a large impact on the distribution parameters, contrary to the situation for global biclustering where the biclusters contain more genes on average. *Variant B* avoids this by removing these background arrays in the first phase of the algorithm.

The 'pure' EM algorithm as described by *Variant A* has relatively poor convergence properties for both global and query-driven biclustering because it suffers from 'race conditions' between the gene-bicluster and the array-bicluster assignments. The process of reassigning both genes and conditions at the same time in the initial phases of the algorithm is highly dependent on the parameters that determine the convergence speed as defined in Section 5.5.5. In case of global biclustering and starting with the complete dataset as initial bicluster, typically more arrays are removed than genes during the first iterations, leading to suboptimal biclusters with many genes and few or none arrays. For query-driven biclustering the problem is less pronounced, but *Variant B* virtually always leads to better local optima.

5.5.7 Time complexity of the EM algorithm

For discussing the time complexity of *Pro*Bic, the following notation is used: G is the number of genes, A the number of arrays, B the number of biclusters and I the number of iterations.

For the maximization step (see Section 5.5.1), a $\mu_{a,b}$ and $\sigma_{a,b}$ value is calculated for every array-bicluster combination. Each calculation is a summation over the number of genes in that bicluster, so an overestimation of the time complexity of this step is: $O(A \cdot G \cdot B)$

Expectation step 1 optimizes the posterior for all $A.B$ combinations and consists of two parts: the first part is the precalculation of all $v_{a,gb,b}$ values and the second part the calculation over all arrays a of the scores for each $a.B$ assignment (see Section 5.5.2). In the first part, the precalculation of the $v_{a,gb,b}$ is of

time complexity $O(A \cdot G \cdot B \cdot log(B))$: the expression value for each array-gene combination is added to one or more (maximum B) $v_{a,gb,b}$ values and the addition itself can be performed in an $O(log(B))$ time with a hash tree of depth B with a gb vector at each of the leaf nodes in that tree.

For the second part, the calculation of the actual score for each array involves an Apriori-like algorithm that generates the candidate $a.B$ vectors (see Section 5.5.2). If the assignment of no more than N biclusters is allowed to change simultaneously, the time complexity of this step is $O(B^N)$. Each of the $a.B$ evaluations (see Equation 5.34) involves a summation over all possible gene-bicluster vector gb and a summation over each bicluster in this vector of some constant values $v_{a,gb,b}$ plus a constant time operation for the second term in Equation 5.34. The summation over each possible gene-bicluster vector gb is $O(B^M)$ if no more than M biclusters can overlap simultaneously. This leads to a total time complexity for the second part of $O(A \cdot B^{N+M})$.

A total time complexity $O(A \cdot G \cdot B \cdot log(B) + A \cdot B^{N+M})$ is obtained for Expectation step 1 by combining the above results. Analogously[3], the time complexity for Expectation step 2 is of order $O(A \cdot G \cdot B \cdot log(B) + A \cdot B^{N+M})$. Assuming that there are I iterations until convergence and that each of the E-steps is followed by an M-step, the time complexity of the complete algorithm is:

$$O(I \cdot (A \cdot G \cdot B \cdot log(B) + (A + G) \cdot B^{N+M})) \tag{5.38}$$

We observe that the *worst case* time complexity is linear in the number of genes and arrays and polynomial in the number of biclusters.

However, for determining the *average* time complexity, we notice that only a small fraction of the biclusters actually overlap in practice. A conservative estimate is that the number of overlaps is of the same order of magnitude as the number of biclusters, so the number of possible gene-bicluster vectors $g.B$ is $O(B)$, leading to an *average* time complexity that is quadratic in the number of biclusters:

$$O(I \cdot (A \cdot G \cdot B \cdot log(B) + (A + G) \cdot B^2)) \tag{5.39}$$

The identification of a single bicluster in a synthetic dataset of 500 genes and 200 arrays, takes about 1 minute on a dual Opteron 250 server with 2 GB RAM (using only one core) using default settings for the convergence speed (see Section 5.5.5).

5.6 Modeling biclusters with anticorrelated profiles

Genes are transcriptionally regulated through either activating, inhibiting or more complex types of interactions. In case of an inhibitory interaction, the

[3]The derivation is not explicitly given, but it is completely analogous to the derivation of E-step 1.

resulting gene expression profile will be anticorrelated with the profile of genes that are regulated through an activating interaction. Most biclustering algorithms do not explicitly identify anticorrelated genes. Instead, researchers often identify two separate biclusters: one with the correlated genes and one with the anticorrelated genes (called *method A* hereafter). Another option is to add the anticorrelated profile of every gene to the microarray dataset and thus effectively duplicating the dataset (*method B*).

Both these methods suffer from drawbacks that deteriorate the quality of the resulting biclusters. *Method A* fails to use the information that both separate biclusters are in fact their anticorrelated counterparts. As a result it may not identify a small set of anticorrelated genes in a bicluster due to noise and other confounding effects that would have been identified if the information regarding their anticorrelation with another bicluster would have been used.

Apart from the additional memory requirements and slower speed for *Method B* that are caused by the duplication of the dataset, *Method B* also deteriorates the quality of the results. The reason for this is more subtle: by adding twice as many genes to the dataset, the probability that spurious biclusters are identified containing only noise has increased compared to the dataset without anticorrelated profiles. Moreover, information is also lost during the data duplication as the identity relation between the anticorrelated and the original gene is broken. From these observations, it is clear that an explicit model for biclusters containing genes with anticorrelated profiles is an interesting property for a biclustering algorithm.

*Pro*Bic is extended to incorporate anticorrelated genes in the bicluster. This can be done in a relatively straightforward way by extending the boolean attributes $g.B_b$ and $a.B_b$ with a third category '-1', indicating the gene or array is part of bicluster b but it is anticorrelated with the other genes. This extension only changes the definition of $f_1(\ldots)$ slightly and it does not affect Equation 5.7. The new definition of $f_1(\ldots)$ becomes:

$$\begin{aligned} & f_1(e.level|g.B, a.B, a.ID) \\ &= f_1^*(e.level|e.array = a, e.biclusters = \{b\}, g.B_b) \\ &= f_1^*(e.level|a, b, g.B_b) \\ &= f_1^*(e.level|\mu_{a,b}, \sigma_{a,b}, g.B_b) \\ &= \frac{1}{\sigma_{a,b}\sqrt{2\pi}} \exp[-\frac{(g.B_b \cdot e.level - \mu_{a,b})^2}{2\sigma_{a,b}^2}] \end{aligned} \tag{5.40}$$

This extension affects the maximization step where the expression values *e.level* of anticorrelated genes need to be negated in case of anticorrelation. The Equations 5.26-5.27 are adapted accordingly. The changes in the Expectation steps are limited to the changed precalculations of the $v_{a,gb,b}$ and $w_{g,ab,b}$ values and the fact that Equations 5.36 and 5.37 are now maximized over the three

possible $g.B_i$ values, namely -1, 0 and 1. Note that, like *method B*, this extension also suffer from an increased probability of identifying spurious biclusters. However, the extension requires less memory and is faster than *method B* and more importantly, the information regarding anticorrelation is explicitly present in the model which is useful if the *Pro*Bic model is extended with additional data sources that are related to the anticorrelated behavior.

5.7 Results

An evaluation of the *Pro*Bic algorithm was performed on simulated gene expression datasets to investigate the behavior of the algorithm under various parameter settings and input data. We tested (1) the robustness of the algorithm w.r.t. noise; (2) the algorithm's ability to handle missing values; (3) the algorithm's ability to incorporate expert knowledge in the form of query genes; (4) the automatic determination of the number of biclusters from the data; (5) the automatic derivation of the optimal parameter settings for a specific dataset; (6) the speed of the algorithm.

A comparison between *Pro*Bic and a number of state-of-the-art biclustering algorithms was performed (Section 5.7.5) and *Pro*Bic has been applied to a number of query-driven settings on a compendium of *E. coli* microarray data (Sections 5.7.6 and 5.7.7).

5.7.1 Datasets

*Pro*Bic has been applied to a number of simulated and biological datasets. The simulated data is generated based on a model assuming biclusters with constant columns (see Figure 5.1 for an illustration). A cross-platform compendium for *E. coli* was used as a biological dataset. In the next Sections, a brief overview is given of these datasets.

Simulated datasets

The simulated data is modeled according to the *Pro*Bic model assumptions. Background values were sampled from a Normal distribution $N(\mu_{bgr} = 0, \sigma_{bgr} = 1)$ for all conditions. The expression values e_{ga} that are part of a single bicluster b were sampled from the distributions $N(\mu_{a,b}, \sigma_{a,b})$ (meaning a separate distribution per condition a and per bicluster b). The expression values in the overlap region between multiple biclusters, were modeled as the average of the values sampled from the distributions $N(\mu_{a,b}, \sigma_{a,b})$ over all overlapping biclusters b. The values for $\mu_{a,b}$ were chosen from a Normal distribution $N(0, \sigma_\mu)$ (excluding the interval $[-\sigma_{\mu_{bgr}}/2, \sigma_{\mu_{bgr}}/2]$) where $\sigma_\mu = 1$ unless otherwise specified. The value $\sigma_{a,b}$ represents the level of 'noise' in the bicluster compared to the background and was defined separately for each of the analyses.

E. coli **compendium**

For the application of *Pro*Bic on biological datasets, a cross-platform compendium for *E. coli* datasets was used. Three major microarray databases contain many of the publicly available microarray datasets on *E. coli*: Stanford Microarray Database (SMD) [42], Gene Expression Omnibus (GEO) [13] and ArrayExpress (AE) [125]. A cross-platform compendium was constructed in by Lemmens et al. [102] that contains a collection of 870 publicly available microarrays for diverse experimental conditions. Both cDNA and oligonucleotide microarray datasets are normalized and combined into a single compendium.

5.7.2 Identification of the number of biclusters

The *Pro*Bic model in principle requires the number of biclusters to be defined upfront. However, the actual number of biclusters in reality is difficult or even impossible to estimate. Firstly, the optimal number of biclusters depends indirectly on the amount of noise in the data. As a model contains more biclusters, also more spurious biclusters will be added that contain a random set of accidently correlated genes. The 'optimal' number of biclusters is the set of biclusters that contains as few spurious biclusters as possible while avoiding to miss any real biclusters. Depending on the research goals, the relative importance of these two numbers (which are in fact the number of false positive and false negative biclusters) will be different and this will be reflected in the parameter settings of the model. Secondly, interesting biclusters exist at different resolution levels [47]. A researcher can both be interested in a small, tightly correlated bicluster or a large, more loosely correlated bicluster that is a superset of the smaller bicluster, depending on his/her research goals. The optimal number of biclusters will therefore necessarily depend on the parameter settings that the researcher applies in a particular study.

This problem recurs in many clustering and biclustering settings [98, 144] and often the number of (bi)clusters is estimated upfront either by the author, e.g. using expert knowledge [144], or by applying other methods that perform an estimation of this number. The *Pro*Bic model can use any of these techniques to estimate the number of biclusters in the dataset. However it can also automatically select the optimal number of biclusters in the dataset for a given parameter setting of the model. The model is initialized with N_{max} (empty) biclusters where N_{max} is chosen larger than the number of actual biclusters in the dataset.

An analysis was performed for simulated 500x200 datasets with varying degrees of noise that were implanted with 7 biclusters each containing 50 genes and 50 arrays. The model score was calculated for models with N biclusters where N varied between 1 and 10. Figure 5.7 shows the model score in function of the number of biclusters in the model for an artificial dataset containing seven biclusters. In an iterative procedure, a model is constructed

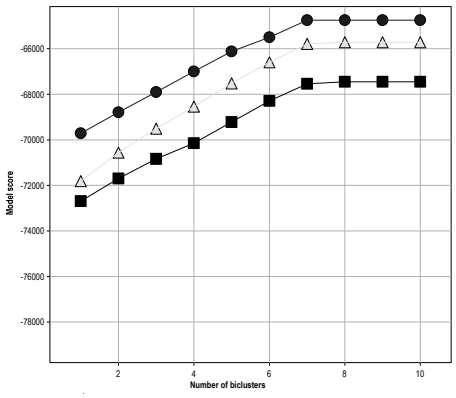


Figure 5.7: ***Identification of the number of biclusters in a simulated dataset for varying degrees of noise.*** *For simulated 500x200 datasets with seven 50x50 biclusters and for varying noise levels, the model score is shown (Y-axis) for a model with N biclusters (X-axis).* Legend noise level: circle: 0.4; triangle: 0.6; square: 0.8.

with 1, 2, . . . optimized biclusters respectively where the remaining biclusters are empty. We observe that the model score increases up until seven identified biclusters, after which the model score does not increase anymore: no more score improvements are achieved since the score would decrease by adding a non-empty bicluster to the model. As soon as the real number of biclusters in the data is obtained, the model will therefore only add empty biclusters to the model.

As discussed, the optimal number of biclusters will vary in function of the different parameter settings. For example, if no penalty is introduced for adding expression values to a bicluster ($\frac{\pi_{bicl}}{\pi_{bgr}} = 1$), the model will select as many biclusters as possible since adding a bicluster increases its degrees of freedom to fit the data.

5.7.3 Optimal model parameter settings

The *Pro*Bic biclustering model has several parameters that potentially affect the biclustering. While it is important for real life applications that parameters can be fine-tuned by the researcher, sensible default values need to be defined that are broadly applicable on a wide range of datasets. In this Section we will investigate optimal parameter settings for each of the parameters of the *Pro*Bic model. Default values for most parameters can be directly derived from the data.

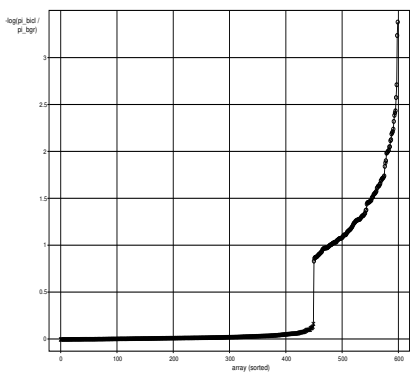
Parameter $\frac{\pi_{bicl}}{\pi_{bgr}}$ ratio

First, how to determine the optimal ratio $\frac{\pi_{bicl}}{\pi_{bgr}}$ that indicates how many times more likely it is for an expression value to be part of a bicluster compared to the background distribution before it is actually added to the bicluster. In order to determine this optimal ratio, we assume there exist one or more known clusters of genes in the dataset that are co-expressed in a subset of conditions. This assumption is not too stringent as in most practical biological situations, such known sets of genes exist (e.g. a set of operon genes). If such a set would not be available, standard clustering techniques can also be used to identify one or more of these clusters.

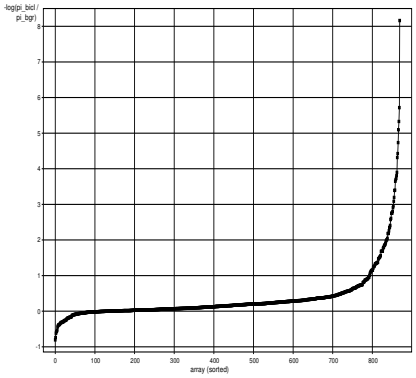
If the conditions are known in which the genes are co-expressed (i.e. we have a set of known *bi*clusters), the optimal ratio we can relatively easy be determined using the following procedure: for every array a, the difference in score (called δ's in the remainder of the section) is calculated between the situations where either that array is part of the bicluster or where it is part of the background. We then solve a classification problem where the optimal threshold is determined for the δ's that minimizes the global error rate (= the product of the false positive rate and the false negative rate) for classifying the arrays in either the background or in the known bicluster. Figure 5.8a shows the δ's for a 500x200 simulated dataset with one known 50x50 bicluster. Applying the above automated procedure leads to the optimal ratio $log(\frac{\pi_{bicl}}{\pi_{bgr}}) = -0.5$.

In case only a set of gene clusters is known but the arrays in which they are coexpressed are unknown, a different procedure can be applied. Again the δ's are calculated for all arrays a. A plot of the sorted δ's is made in which a clear cutoff point between arrays with high and low δ's can often be determined visually, as can be seen in Figure 5.8b. This Figure shows the δ's for all arrays in the *E. coli* compendium for the set of genes that are known to be regulated by *FNR*. Based on this plot, good cutoff values for $-log(\frac{\pi_{bicl}}{\pi_{bgr}})$ would range between 0.5 and 0.8. *FNR* is a global regulator and therefore no tight bicluster exists that covers all *FNR* regulated genes. Therefore there is a smooth transition between bicluster and background conditions and no unique cutoff value can be determined. A broader region can however be defined that separate bicluster and background conditions. An alternative to selecting a single global regulator would be to select a large set of specific regulators and define their target sets as the set of gene clusters for which the δ's are plotted. The latter approach would also average the selection of the cutoff value over a large number of gene sets, thereby increasing the robustness of the method.

Next we investigate the sensitivity of the bicluster inference quality with respect to the $\frac{\pi_{bicl}}{\pi_{bgr}}$ ratio. Figure 5.9 shows how precision and recall for the arrays vary with respect to the $\frac{\pi_{bicl}}{\pi_{bgr}}$ ratio for a synthetic 500x200 dataset with three non-overlapping 50x50 biclusters for a number of different noise levels. As



(a) $delta_{score}$ for a simulated dataset



(b) $delta_{score}$ for all known FNR regulated genes in the *E. coli* compendium

Figure 5.8: ***Illustration of determining the optimal ratio for*** $\frac{pi_{bicl}}{\pi_{bgr}}$.
(a) *For a synthetic 500x200 dataset with three 50x50 biclusters (noise level 0.2). The plots show the* $delta_{score}$ *over all arrays (multiplied with the number of biclusters) for a set of genes that are 'known' to be co-expressed in a number of arrays. Large* $delta_{score}$*'s indicate that the genes are likely to be in the bicluster distribution for that condition (compared to the background distribution). The* $delta_{score}$ *threshold that classifies these two sets of arrays best according to the ratio is 0.5, leading to an optimal value of* $-log(\frac{\pi_{bicl}}{\pi_{bgr}}) = 0.5$. ***(b)*** *Sorted* $delta_{score}$ *levels for all FNR regulated genes over all arrays in the E. coli compendium. Based on this plot, good values for* $-log(\frac{\pi_{bicl}}{\pi_{bgr}})$ *range between* 0.5 *and* 0.8.

the $\frac{\pi_{bicl}}{\pi_{bgr}}$ ratio decreases, the precision is very high (precision=1) since only real bicluster arrays are kept in the identified bicluster. Such high precision comes however at the expense of a lower recall. The optimal predicted $log(\frac{\pi_{bicl}}{\pi_{bgr}})$ values for the simulated datasets were -0.44, -0.36 and -0.22 for noise levels 0.2, 0.5 and 1.0 respectively. Figure 5.9 shows that for these parameter values a precision and recall of almost 100% is achieved.

Parameter β

The overall probability that a gene is in any bicluster (β) indirectly affects the size (in number of genes) of the biclusters. Figure 5.10 illustrates the effect of varying β on the bicluster size for a 500x200 dataset with one 100x100 bicluster that is composed out of three parts:

- expression values for 20 genes, sampled from $N(\mu_a, 0.2)$
- expression values for 30 genes, sampled from $N(\mu_a, 0.5)$
- expression values for 50 genes, sampled from $N(\mu_a, 1.0)$

We expect that lower β values will lead to a smaller bicluster (keeping only the most correlated genes) and this is confirmed in Figure 5.10. A single bicluster is identified with the *Pro*Bic model and the number of genes in the bicluster decreases successively from 100 to 50 to 20 for decreasing β as predicted. We confirmed that the identified bicluster indeed corresponded to the original 100x100, 50x100 and 20x100 biclusters respectively.

5.7.4 Noise and missing values robustness

To assess the effects of noise and of missing values on the bicluster identification, a 500x200 (genes x arrays) simulated dataset was constructed as previously described with three 50x50 non-overlapping biclusters. The percentage of missing values in the dataset varied between 0% and 100% and the bicluster noise level was varied between 0 (no bicluster noise) and 1 (bicluster noise = background noise). The resulting effect on the precision and recall for both the genes and the arrays is shown in Figure 5.11.

Figures 5.11a and 5.11c show that for lower bicluster noise levels (< 0.5), the presence of up to 60% of missing values in the dataset does not interfere with the detection of the true bicluster genes and arrays (a precision and recall of about 100% is obtained for both the genes and arrays). As can expected the model sensitivity to missing values increases with the bicluster noise level, but even in the presence of high noise levels (1.0), the presence of up to 20% missing values does not considerably deteriorate the algorithms recall and precision (recall levels of 0.9 and 0.88 for the genes and arrays respectively).

5.7.5 Comparison of *Pro*Bic with state-of-the art query-driven biclustering algorithms

Since many biclustering algorithms are based on different statistical measures of 'good' biclusters, a fair comparison between the algorithms is difficult and often biased. Rather than using the simulated data described in Section 5.7.1, we used previously published artificial data to avoid any bias in the results. In Dhollander et al. [47], a systematic comparison of query-driven biclustering algorithms was performed using various simulated gene expression datasets defined by Prelic et al. [131]. The tested algorithms were ISA [79, 78], Gene Recommender [123] and QDB [47].

Prelic et al. [131] have investigated various experimental settings. Datasets were generated containing (1) biclusters with constant expression values, called *scenario 1*, and (2) datasets containing biclusters with constant columns and an additive bicluster model, called *scenario 2*. The datasets are 100 genes by 50 arrays (scenario 1) and 100 genes by (100...110) arrays (scenario 2) respectively. For each of these scenarios, two settings were tested. In *setting A*, noise is added to the gene expression values and a model without overlap between the biclusters is used. The standard deviation σ of this distribution

represents the noise level. In *setting B* an increasing amount of overlap between the biclusters is tested, varying between 0 (no overlap) and 10 (50% overlap) for both genes and arrays of the biclusters. In each dataset 10 biclusters are implanted of fixed size 10x10 for *setting A* and of an increasing size $(10 + i)x(10 + i)$ in *setting B* where i represents the amount of overlap. More details on the experimental setup can be found in [131].

If we assume two *sets* of biclusters M_1 and M_2 and we denote the set of genes of an individual bicluster $B_i = (G_i, C_i)$ with G_i and the set of conditions with G_i, then the following *gene match score* is used to asses the performance of the algorithms, as defined in Prelic et al. [131]:

$$S^*_G(M_1, M_2) = \frac{1}{|M_1|} \sum_{G_1, C_1 \in M_1} max_{G_2, C_2 \in M_2} \frac{|G_1 \cap G_2|}{|G_1 \cup G_2|}$$

This score lies between 0 and 1 and it is 1 if the bicluster sets are identical and the score is 0 if none of the gene sets of the bicluster sets show any overlap.

If we define M_{real} as the true set of implanted biclusters in the dataset, then the average *bicluster relevance score* for a bicluster M is defined as $S^*_G(M, M_{real})$ and the average *bicluster recovery score* as $S^*_G(M_{real}, M)$. The bicluster relevance reflects to what extent the identified biclusters represent true biclusters in the gene dimension and the bicluster recovery determines how well the true biclusters are recovered by the algorithm.

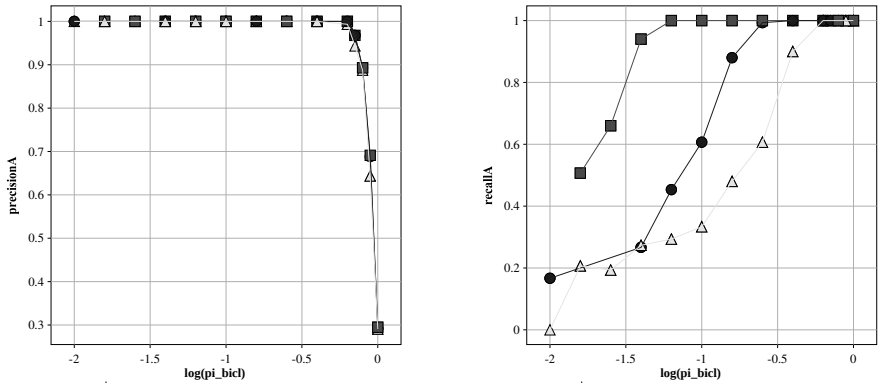
Figure 5.12 shows the results of this analysis. In general, all query-driven algorithms perform very well on these data compared to various global biclustering methods. We refer to Prelic et al. [131] for a detailed analysis of global biclustering algorithms on these datasets and focus here instead on the results for query-driven biclustering.

For the noise experiments (*setting A*), we observe that *Pro*Bic performs well for low noise levels but it is more sensitive to higher noise levels compared to the other biclustering algorithms. These results are in contrast with the noise and missing value robustness results obtained in Section 5.7.4. In order to verify if indeed noise was the cause of this reduced performance, an additional series of experiments was performed using *Pro*Bic global biclustering (PB-G) on the simulated datasets and the results for scenario 1 are added to Figure 5.12. The global biclustering approach leads to a perfect recovery for all cases in scenario 1. However poor recovery and relevance were observed for scenario 2 ($score < 0.1$, results not shown).

The fact that the global biclustering recovers biclusters that are not identified in the query-driven setting was not due to any differences in the initialization and a resulting local optimum of the EM algorithm: experiments for query-driven biclustering with different types of initialization (complete dataset and random initializations) lead to the same (global) optima. The reason for the poor performance on noisy datasets of query-based clustering is due to our

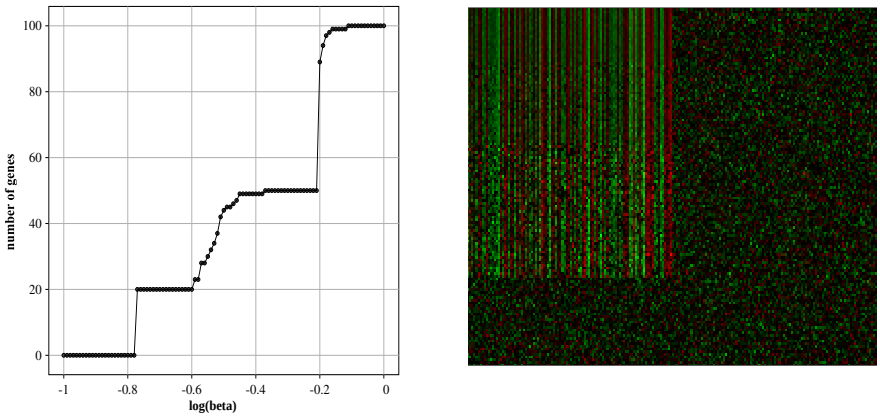
choice of prior distribution and its parameters (see Section 5.5.4). For the query-driven biclustering a Normal-Inverse-χ^2 prior was used in the *Pro*Bic model. Analyses have shown that the performance of *Pro*Bic is sometimes greatly affected by the choice of prior distribution and its associated parameters: e.g. up to 20% variation for the bicluster relevance and recovery scores were seen in *setting A* for varying the σ_0 parameter which controls how 'tight' the bicluster profile should be *a priori*. All *Pro*Bic results shown in Figure 5.12 are for a choice of $\sigma_0 = 0.4\sigma_{bgr}$. While *Setting A* was very sensitive to these parameter settings, *Setting B* was more robust for variations in the parameters of the prior distributions.

Although this analysis on simulated data is helpful to gain insight in the characteristics of the algorithm, it is however important to realize some shortcomings of these datasets. Realistic datasets contain thousands of genes and several hundreds of conditions. Due to computational limitations for performing the comparison analysis, the simulated datasets are much smaller in size. Secondly, the modules in the simulated data are all of nearly equal size and vary relatively little with respect to noise compared to real biological datasets. Thirdly, other definitions of overlap could be defined than the additive overlap model that was used here. Finally, there are no real 'background' genes in these datasets that are not part of any bicluster. The latter leads to a disadvantage for biclustering methods with an explicit background model such as QDB and *Pro*Bic since the high percentage of biclusters confounds a robust background estimation.



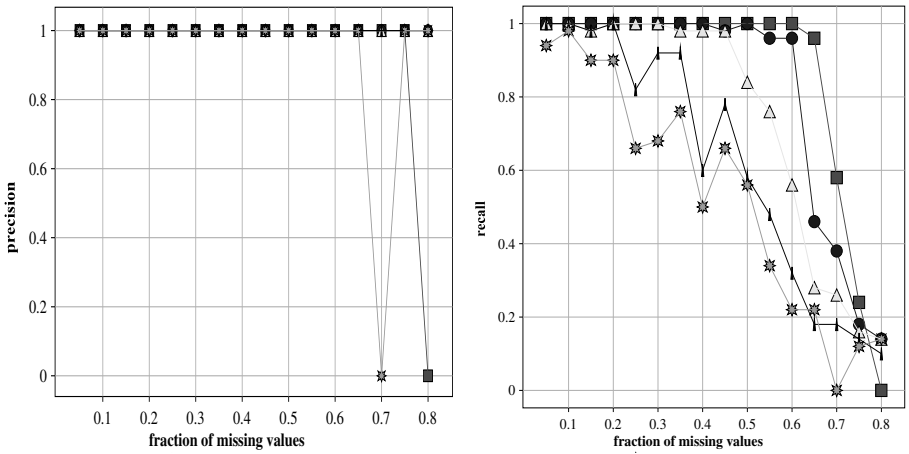
(a) precision (arrays) (b) recall (arrays)

Figure 5.9: ***Sensitivity of the array precision and recall w.r.t.*** $\frac{\pi_{bicl}}{\pi_{bgr}}$***.*** *The effect is shown of varying* $log(\pi_{bicl})$ *on the precision and recall for the arrays for a 500x200 synthetic dataset with three non-overlapping 50x50 biclusters for varying levels of noise (square: 0.2; circle: 0.5; triangle: 1.0).* π_{bgr} *is considered 1 here since only the ratio* $\frac{\pi_{bicl}}{\pi_{bgr}}$ *is important.*

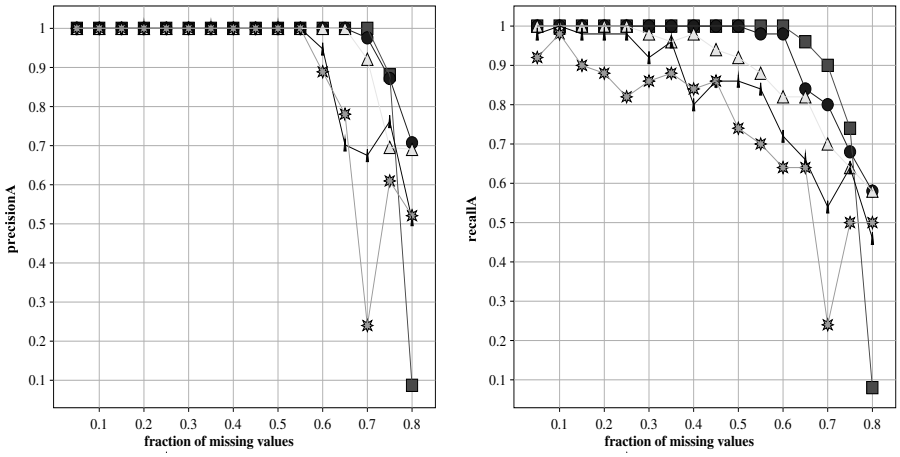


(a) Effect of β on the number of genes. (b) Heatmap of the simulated dataset.

Figure 5.10: ***Illustration of the effect of the*** β ***parameter on the number of inferred genes in a bicluster****. A 500x200 simulated dataset contains one 100x100 bicluster that is composed out of three parts: the expression values for 20 genes are sampled from* $N(\mu_a, 0.2)$ *distributions, for 30 genes from* $N(\mu_a, 0.5)$ *and for 50 genes from* $N(\mu_a, 1.0)$*.* ***(a)*** *Effect of the* $log(\beta)$ *value on the number of genes in the inferred bicluster.* ***(b)*** *Heatmap plot of a part of the simulated dataset containing the 100x100 bicluster with varying degrees of noise its expression values.*

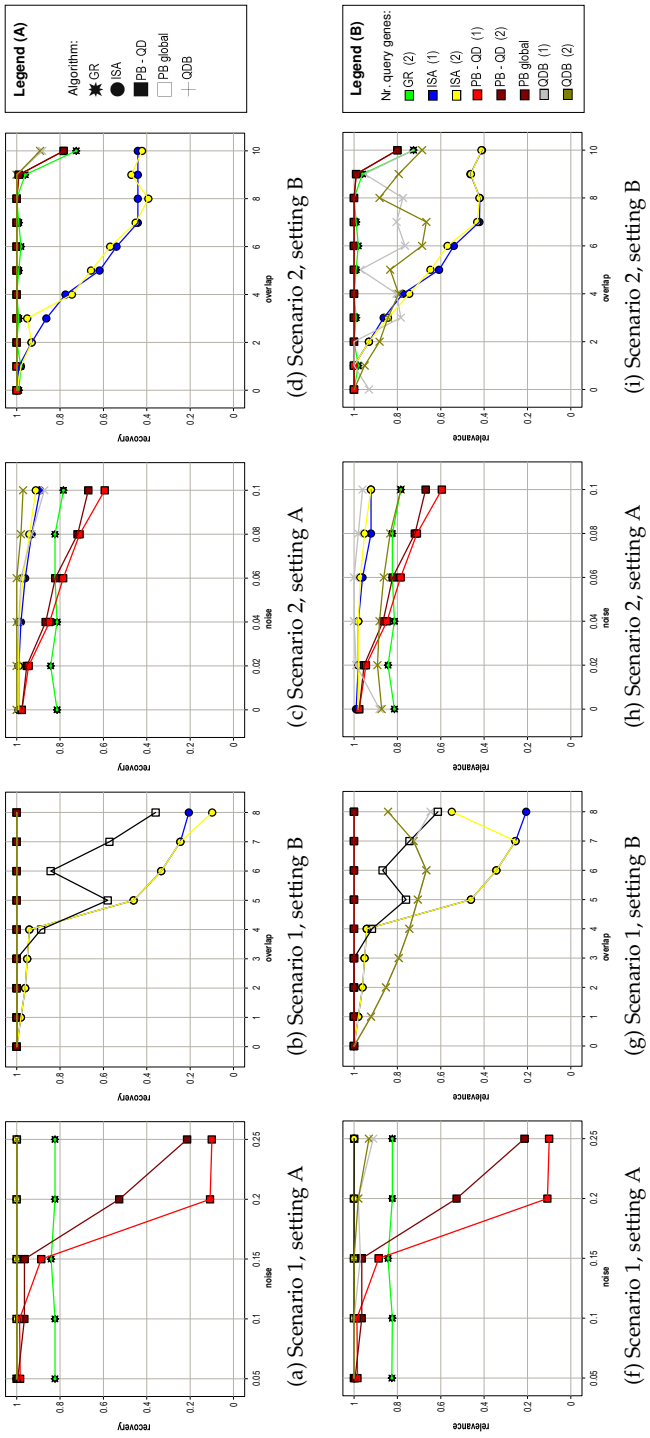


(a) Precision genes (b) Recall genes



(c) Precision arrays (d) Recall arrays

Figure 5.11: ***Noise and missing value robustness of*** **Pro*Bic***. Pro*Bic performance on a 500x200 simulated dataset (containing three 50x50 biclusters) with varying degrees of noise and missing values. The X-axes show the percentage of missing values (between 0% and 100%) in the dataset and the plots indicate the average precision (= $\frac{TP}{TP+FP}$) and recall (= $\frac{TP}{TP+FN}$) for the genes and arrays respectively.* Noise level legend: square: 0.2; circle: 0.4; triangle: 0.6; small triangle: 0.8; star: 1.0.



(a) Scenario 1, setting A (b) Scenario 1, setting B (c) Scenario 2, setting A (d) Scenario 2, setting B

(f) Scenario 1, setting A (g) Scenario 1, setting B (h) Scenario 2, setting A (i) Scenario 2, setting B

Figure 5.12: ***Performance comparison of* ProBic *vs. other query-driven biclustering algorithms for simulated datasets described in Prelic et al. [131].*** ProBic *(PB-QD) is compared to Gene Recommender [123] (GR), ISA [19] and QDB [47]. The bicluster recovery scores are shown in (a)-(d) and the bicluster relevance scores in (f)-(i). Figures (a), (c), (f) and (h) these scores are plotted in function of the amount of noise and in (b), (d), (g) and (i) they are plotted in function of the overlap degree.*

5.7.6 Query-driven biclustering with single gene queries

Some query-driven biclustering algorithms such as Gene Recommender [123] are known to deal badly with single gene queries. Therefore, a number of experiments were conducted to identify biclusters in the *E. coli* compendium based on single gene queries. Table 5.2 shows a selection of six known target genes of different regulators together with the identified biclusters, known regulator and gene ontology information. For verifying that the algorithm converges to similar biclusters for different target genes of known regulators, two different seed genes were chosen for both the *LexA* and *CysB* regulators. The identified biclusters are shown in Figure 5.13.

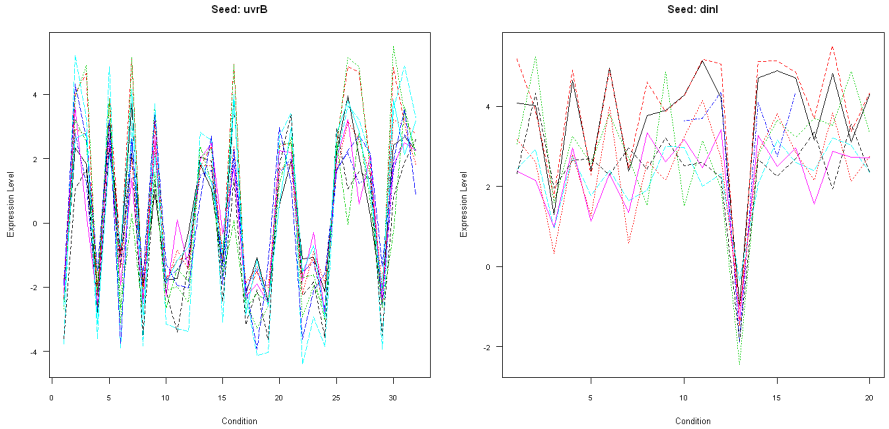
Reg.	Query	Genes	Arrays	Reg. enrich.	GO term
LexA	*uvrB*	11	32	*LexA* (1.0394e-14)	SOS response (1.13e-19)
LexA	*dinI*	8	20	*LexA* (9.8312e-06)	SOS response (2.05e-12)
Fur	*fhuE*	20	75	*Fur* (1.3682e-23)	enterochelin (enterobactin) (1.43e-12)
CysB	*cysK*	12	97	*CysB* (2.3838e-20)	Sulfur metabolism (5.36e-12)
CysB	*cysD*	10	110	*CysB* (1.0744e-18)	unknown
NtrC	*ddpX*	7	28	*NtrC* (7.0782e-10)	nitrogen metabolism (5.04e-02)

Table 5.2: Overview of the used seed genes for query-driven biclustering with single gene queries. For each query gene, the known regulator is shown together with the number of genes and arrays in the identified bicluster. An enrichment score was calculated for all known regulator target genes based on RegulonDB data and the most enriched regulator is shown for each module together with the p-value. Finally, the most enriched gene ontology term is shown together with its associated p-value.

5.7.7 Outlier removal for query-driven biclustering

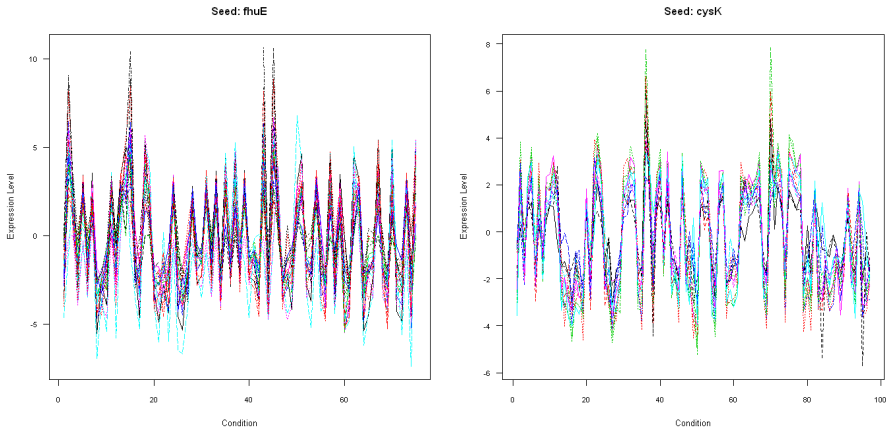
Biologists are often interested in the genes that are part of a specific pathway for which only a subset of the genes is known. In some cases, the set of query genes contains one or more *outliers*, genes that actually do not belong to the pathway. In such cases, the query-driven biclustering should remove these outlier genes from the query while retaining the other genes in the final bicluster that is associated with the pathway of interest.

We simulated this situation for the *E. coli* compendium, by selecting two co-expressed query genes that belong to the *cyoABCDE* operon *cyoA* and *cyoB*. The resulting bicluster is shown for this query in Figures 5.14a-b. Two settings with outlier genes were investigated with respectively one and two additional



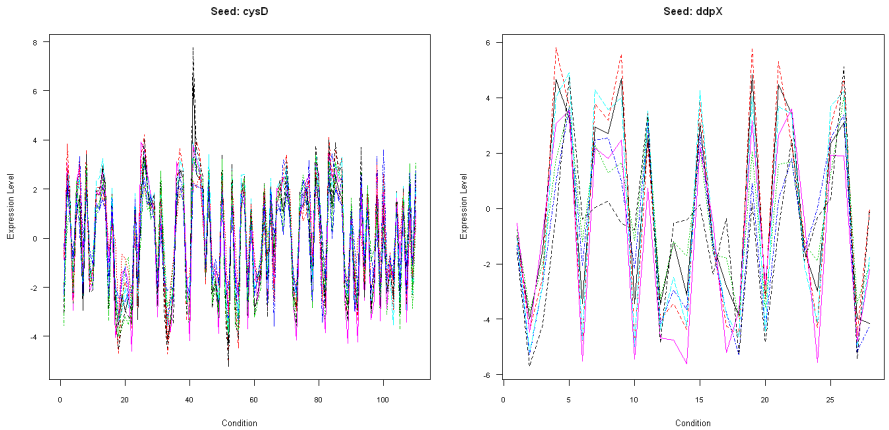
(a) *uvrB*

(b) *dinI*



(c) *fhuE*

(d) *cysK*



(e) *cysD*

(f) *ddpX*

Figure 5.13: ***Identified biclusters expression profiles for different single gene queries.***

randomly chosen genes, namely (*rfbB*) and (*rfbB*, *yniB*). The resulting biclusters are almost identical in both genes and conditions, as shown in Figures 5.14c-d. The results show that even when 40% noisy query genes, the correct bicluster is still identified and the noisy genes are removed from the resulting bicluster.

All resulting modules were most enriched for the *Fur* regulator. *Fur* is an inhibitor which has been implicated in the regulation of a large number of operons that encode enzymes involved in iron transport and is a known regulator for the *cyoABCDE* operon. *Fur* *p*-values ranged between 1e-29 and 1e-16, except for query (*cyoA*, *cyoB*, *rfbB*) where both *Fur* (p=1e-16) and *ArcA* (p=4e-17) were most enriched.

5.8 Extending the *Pro*Bic model

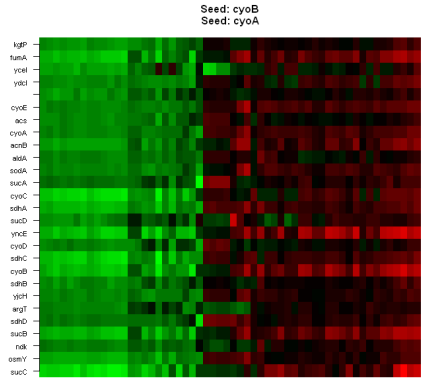
The following Sections describe a number of extensions to the *Pro*Bic model towards the integration of heterogeneous data. The combination of the PRM framework with an EM optimization strategy lead to a powerful and modular approach for building integrative models that identify transcriptional regulatory modules.

5.8.1 Integration of sequence data

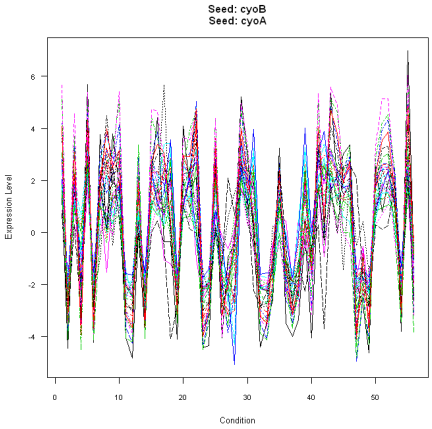
The *Pro*Bic model can be extended towards identification of cis-regulatory modules by incorporating promoter sequence data, similar to the approach described by Segal et al. [147] for the identification of gene clusters and the associated regulatory motifs. The following extension can both be considered as an extension of *Pro*Bic by incorporating promoter sequences and as an extension of Segal et al. [147] by biclustering over both gene and arrays rather than gene clustering only.

In Figure 5.15, a graphical illustration is shown of the resulting PRM for such an extension. An additional class *Promoter* is introduced that contains the promoter sequences $p.S$ for all the genes. The boolean variables $g.R_r$ are hidden and indicate if a gene is being regulated by a specific regulator r. They are conditionally dependent on the promoter sequence and are modeled using a position specific scoring matrix (PSSM). Note that the determination of the MAP values of these attributes will depend both on the promoter sequence data of the gene and the associated gene-bicluster attributes $g.B$. Each bicluster in this model is now associated with one or more regulators, which is modeled by the CPD $P(g.B|g.R)$. In Segal et al. [147] this is for example modeled using a *softmax* distribution. In the proposed model, this CPD associates a regulator profile to each bicluster.

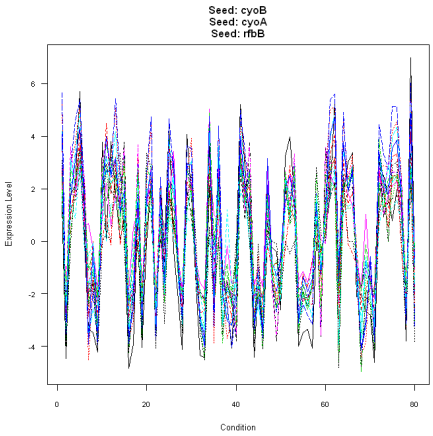
The JPD for the proposed model is given by Equation 5.41:



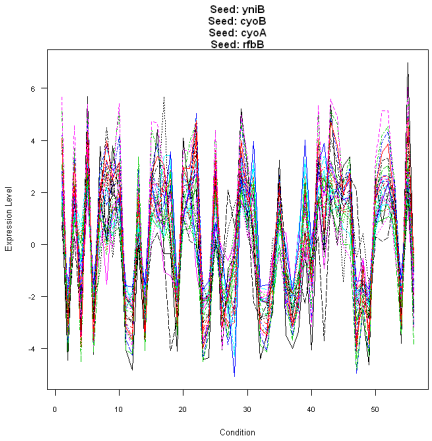
(a) Bicluster heatmap (sorted arrays) for query (*cyoA*, *cyoB*)



(b) Bicluster profile for query (*cyoA*, *cyoB*)



(c) Bicluster profile for query (*cyoA*, *cyoB*, *rfbB*)



(d) Bicluster profile for query (*cyoA*, *cyoB*, *rfbB*, *yniB*)

Figure 5.14: ***Illustration of the effect of a 'noisy query' for a selected in the*** *E. coli* ***compendium****. (a) and (b) show the heatmap and expression profile for the identified bicluster using two co-expressed query genes* cyoA *and* cyoB *that belong to the* cyoABCDE *operon. (c) and (d) show the resulting bicluster profiles for noisy queries with one (*rfbB*) and two (*rfbB *and* yniB*) additional random genes respectively to the original query* cyoA *and* cyoB*.*

$$
\begin{aligned}
likelihood &= P(E.L, G.B, G.R, P.S, A.B) \\
&= P(E.L|G.B, A.B, A.ID) \cdot P(A.B) \cdot P(G.B|G.R) \cdot P(G.R|P.S) \cdot P(P.S) \\
&= \prod_{e \in E} P(e.L|g.B, a.B, a.ID) \cdot \prod_{a \in A} \prod_{k \in K} P(a.B_k) \cdot \\
&\quad \prod_{g \in G} \prod_{b \in B} P(g.B_b|g.R) \cdot \prod_{g \in G} \prod_{r \in R} P(g.R_r|P.S) \cdot \prod_{p \in P} P(p.S) \qquad (5.41)
\end{aligned}
$$

The MAP solution of this model is also determined using an EM approach similar to the *Pro*Bic EM algorithm. Because of the design of the model extension, namely that expression values depend only on the gene- and array-bicluster assignments $g.B$ and $a.B$, the existing EM steps as developed for *Pro*Bic remain unchanged in the extended model. The Expectation step is now split into three substeps: the two existing *Pro*Bic steps and one additional E step that maximizes the posterior w.r.t. the hidden variables $g.R$. The Maximization step of *Pro*Bic is extended with two additional and independent optimization parts: the first part optimizes the PSSMs for defining the regulatory motifs and the second part maximizes the softmax parameters.

A detailed description of how these extensions are modeled can be found in [147] and in the PhD thesis of Segal [143] as part of the model description of the identification of cis-regulatory modules.

5.8.2 Integration of microarray condition property data

Recently, large microarray datasets have become available with annotated condition properties [102]. Such annotation includes all information regarding the experimental conditions, for example the concentrations of all compounds in the medium, either as absolute values or as relative changes compared to a reference experiment. The use of such condition annotation data can lead to novel insights about which biclusters or regulatory modules are associated with specific (changes in) experimental conditions. E.g. a bicluster regulated by a transcription factor that is associated with DNA repair might be strongly associated with experimental conditions that have high concentrations of a toxic compound X that damages DNA.

The *Pro*Bic model can be extended in an analogous way to Section 5.8.1, but rather than the genes, the arrays are now extended with a number of additional attributes, namely the condition annotations as indicated with $a.C_i$ in Figure 5.16. These condition annotations are either continuous (e.g. compound concentrations) or discrete (e.g. gene knockouts) and they are associated with a bicluster through the CPD $P(A.B|A.C)$. A softmax distribution can also be used here similar to Section 5.8.1 and the *Pro*Bic EM algorithm can be extended analogously to include the condition annotation data.

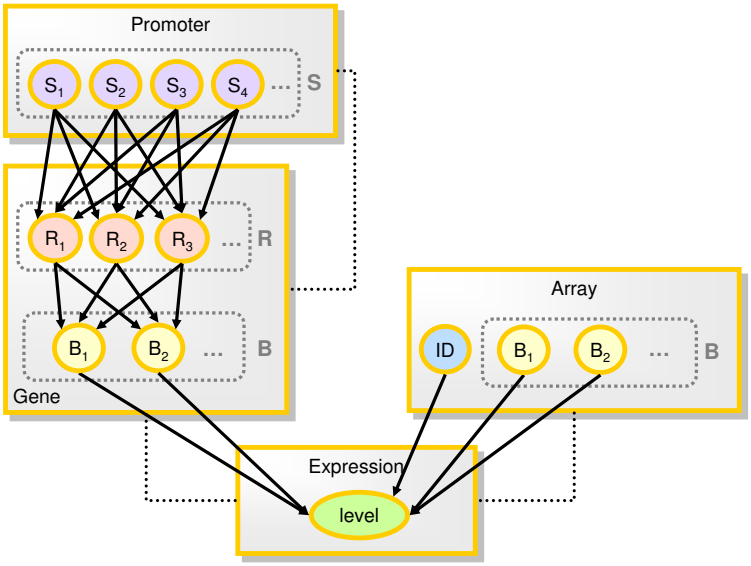


Figure 5.15: ***Extension of the ProBic model towards identification of cis-regulatory modules.***

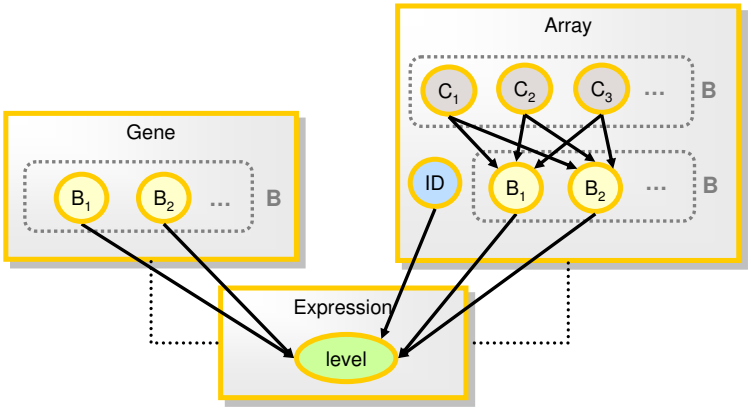


Figure 5.16: ***Extension of the ProBic model by incorporating condition annotation data.***

5.8.3 Identification of regulatory modules with condition annotation data

The full power of using PRMs to integrate relational data in combination with an EM strategy can be illustrated by combining the extensions described in Sections 5.8.1 and 5.8.2 into an integrative probabilistic model that identifies regulatory modules with the associated regulatory motifs, experimental conditions and associates these modules with one or more specific condition annotations. Figure 5.17 visualizes the resulting PRM. Due to the particular way that these model extensions preserve the independence relations between the existing attributes in the *Pro*Bic model and the extensions, this combined model can again use the unmodified Expectation and Maximization substeps as defined in Sections 5.8.1 and 5.8.2.

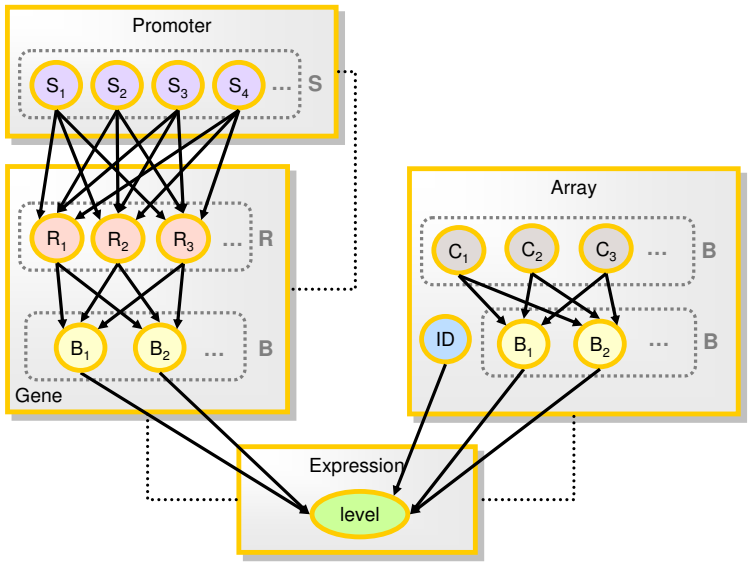


Figure 5.17: ***Extension of the* ProBic *model incorporating both promoter sequence and condition annotation data.***

5.9 Discussion

Design of the *Pro*Bic model

The *Pro*Bic model has been designed without an explicit *Bicluster* class in the model. At first sight, it may seem counterintuitive for not explicitly modeling this class. However, the design of a PRM with a separate *Bicluster* has some disadvantages as we will outline. A possible structure of a PRM with a separate

bicluster class is shown in Figure 5.18 and the JPD for this model is given by Equation 5.42.

$$\begin{aligned} likelihood &= P(E.L, M.B, A.ID) \\ &= P(E.L|M.B, A.ID) \cdot P(M.B) \cdot P(A.ID) \\ &= \prod_{e \in E} P(e.L|e.module.B, e.array.ID) \cdot \\ &\quad \prod_{m \in M} P(m.B) \cdot \prod_{a \in A} P(a.ID) \end{aligned} \tag{5.42}$$

In a model with an explicit *Bicluster* class (named M in Equation 5.42), 'biclusters' are now defined at the expression level rather than at the gene or array level. This means that in principle the shape of the biclusters is not restricted anymore to rectangular blocks. Any set of expression levels can be grouped in a bicluster in principle, the only factor that can prevents this from occurring is the expression level CPD. Such models may however prove to be useful for other domains such as e.g. image analysis applications.

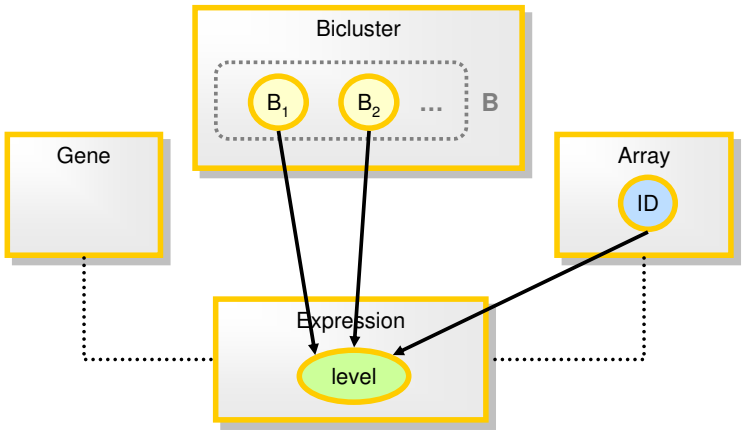


Figure 5.18: ***Alternative PRM model for biclustering gene expression data with an explicit*** *Bicluster* ***class.***

The structure of this PRM also does not allow for the types of priors on the gene-bicluster assignments as defined in Section 5.4 since the bicluster attributes are no longer defined at the gene level. The proposed *Pro*Bic extensions from Section 5.8 can not be modeled in a straightforward way in a PRM with an explicit *Bicluster* class for the same reason: the bicluster attributes are no longer defined at the gene and array level.

In conclusion, while a biclustering model with a separate bicluster class would be intuitive from a data-centric point of view, such model has undesired prop-

erties from a model-centric point of view. These considerations not only apply to *Pro*Bic but are more generally applicable in the context of relational models. It is often necessary to reshape the relational structures in the dataset in order to achieve the desired modeling properties such as the decomposability and the proper definition of the CPDs towards their underlying real-world model.

Different overlap models

For the *Pro*Bic model, a particular overlap model was implemented that averages the individual probabilities of the expression level belonging to each of the overlapping biclusters. In principle many other types of overlap models can be chosen as the described overlap model. Depending on the biological process of interest and properties of the gene expression data such as the normalization procedure, other overlap models can be defined such as a sum, average or weighted average overlap models as defined in Equations 5.44-5.45, with γ_b the weights of each of the biclusters b, that are of biological interest to the researcher:

$$\begin{aligned} & f_1^{sum}(e.level|e.gene.B, e.array.B, e.array.ID) \\ &= f_1^*(e.level|\mu = \sum_{\substack{b\in \\ iset(B_e^i)}} \mu_{a,b}; \sigma = \sqrt{\sum_{\substack{b\in \\ iset(B_e^i)}} \sigma_{a,b}^2}) \end{aligned} \tag{5.43}$$

$$\begin{aligned} & f_1^{avg}(e.level|e.gene.B, e.array.B, e.array.ID) \\ &= f_1^*(e.level|\mu = \frac{1}{\#iset(B_e^i)} \sum_{\substack{b\in \\ iset(B_e^i)}} \mu_{a,b}; \sigma = \sqrt{\sum_{\substack{b\in \\ iset(B_e^i)}} \sigma_{a,b}^2}) \end{aligned} \tag{5.44}$$

$$\begin{aligned} & f_1^{weighted}(e.level|e.gene.B, e.array.B, e.array.ID) \\ &= \frac{\prod_{\substack{b\in \\ iset(B_e^i)}} f_1^*(e.level|\mu = \mu_{a,b}; \sigma = \sigma_{a,b})^{\gamma_b}}{\int_{e=-\infty}^{+\infty} \prod_{\substack{b\in \\ iset(B_e^i)}} f_1^*(e.level|\mu = \mu_{a,b}; \sigma = \sigma_{a,b})^{\gamma_b} de} \end{aligned} \tag{5.45}$$

Extensions to the *Pro*Bic model

Extensions to the *Pro*Bic model are always confined to the implicit restrictions that apply for the PRM framework. Firstly, the resulting ground Bayesian network associated with the PRM and a dataset needs to be a DAG. This requirement is automatically fulfilled if the PRM model itself is a DAG (note however that this is an overly restrictive condition). Extensions that lead to cycles in the GBN are therefore not allowed. Secondly, a PRM is a *directed* graph, so undirected relations between attributes can not be modeled within the PRM framework. The model can be extended to more general statistical relational modeling frameworks, but at the cost of convergence guarantees and other statistical properties that can be guaranteed for PRMs.

Thirdly, while the convergence properties with respect to identifying the global optimum are fairly good for the current *Pro*Bic model, the extension of the model with additional attributes or datasets may adversely influence the algorithm's convergence. It is difficult to assess a priori what the effect will be of such a particular model extension. Additionally, also the particular implementation of the EM algorithm affects the convergence. Therefore, any model extension needs to be validated with a new convergence assessment on either simulated or biological datasets.

5.10 Summary

We have developed a novel biclustering model, called *Pro*Bic, that builds upon the expertise of previous work and extends the current generation of biclustering methods in the following areas: (1) *Pro*Bic uses a model-based approach for identifying multiple biclusters simultaneously; (2) both global and query-driven biclustering can be performed within the model, even simultaneously; (3) *Pro*Bic groups both correlated and anticorrelated genes within a bicluster (diametrical biclustering) and is, to our best knowledge, the first constant column biclustering algorithm with diametrical biclustering that has been developed; (4) *Pro*Bic is based on continuous expression data, so no discretization of the gene expression data is required; (5) for query-driven biclustering, seed genes can be removed from the resulting bicluster if they have no matching expression profile with the bicluster, a feature that is important in practice to biologists.

An Expectation-Maximization algorithm has been developed for learning the model parameters and hidden variables efficiently. Experiments on both simulated data show the robustness of the model with respect to noise and missing values. Default parameter settings are automatically determined for each specific dataset, while retaining the possibility of fine-tuning these parameter settings to the researcher's needs.

Results on simulated data from Prelic et al. [131] show that *Pro*Bic competes with state-of-the-art biclustering algorithms and even outperforms ISA and QDB for settings with high overlap between biclusters. A robustness analysis with respect to noise and missing values on simulated data indicated that *Pro*Bic can deal with relatively high noise and missing value levels, a necessary feature for biclustering inherently noisy microarray datasets. In query-driven settings where the set of query genes contain one or more outliers, the algorithm also performs well and is able to remove such outlier genes from the identified bicuster. This feature is particularly useful to biologists in practice.

*Pro*Bic is embedded in the relational framework of PRMs and therefore allows for integrative extensions of the biclustering model towards the identification of regulatory modules by incorporating additional data sources in a unified

probabilistic model. Several extensions have been outlined that integrate both sequence data and microarray condition property data. These model extensions and their combinations reveal the power of using PRMs in combination with an Expectation-Maximization approach as the existing EM steps can be used unmodified in the extended models.

Chapter 6

Conclusion

In this Chapter we summarize the main contributions of this thesis and we provide some directions for future work. The research described in this thesis covers a number of topics related to inferring regulatory networks based on gene expression data.

6.1 Summary and achievements

The first part of this thesis presents the development of a simulator for transcriptional regulatory networks, called SynTReN (Figure **??**), that generates the associated gene expression data sets for the generated network. Three well known network inference algorithms have been applied to various settings of the simulated data and have led to insights in the operational characteristics of these algorithms that would have been difficult or impossible to achieve using biological data only.

- We designed a gene network generator and simulator that is able to simulate large regulatory networks with thousands of genes. Current state-of-the-art dynamic simulators that simulate networks up to maximally a few hundred genes. By reducing the data generation to only steady-state solutions, the simulation of a network comprising thousands of genes becomes computationally tractable.
- Our results show that the choice of network topology for the simulated data can profoundly influence the quality of the results of some inference algorithms. While inference algorithms are often tested on simulated data, the topology of the underlying network is usually not considered as key factor. Disregarding this aspect leads to biased and possibly faulty conclusions based on the results on simulated data.

- Different inference algorithms were applied to simulated datasets with various characteristics. The results show a qualitatively very different response of the algorithms with respect to parameters of the simulated data such as noise, amount of data, types of interactions. These results also prove that simulated data is useful to provide more insights in the operational characteristics of an algorithm that are complementary to the insights gained from experiments on real biological data and that are unlikely to be discovered by means of biological data only.

The second part of this thesis describes the development of a model-based biclustering algorithm that is developed within the framework of probabilistic relational models. Both query-driven and global biclustering can be combined in a single model and the model simultaneously identifies multiple potentially overlapping biclusters containing genes with both correlated and anticorrelated profiles:

- An efficient biclustering algorithm, called *Pro*Bic, has been developed within the framework of probabilistic relational models, requiring no prior discretization of the gene expression data.
- The biclustering model naturally deals with missing values and noise due to its probabilistic nature. This leads to robust identification of biclusters under various settings of noise and missing values.
- Both global and query-driven biclustering can be combined within a single model-based approach and the query-driven biclustering has been proven robust with respect to outlier genes in the set of *seed genes*.
- *Pro*Bic simultaneously identifies multiple overlapping biclusters.
- An extension to *Pro*Bic allows to group both correlated and anticorrelated genes within a single bicluster.
- The powerful combination of PRMs with an Expectation-Maximization approach allows *Pro*Bic to be easily extended to incorporate additional data sources, ultimately leading to the identification of regulatory modules with associated condition annotation, regulatory motifs and transcription factors.

6.2 Future work

SynTReN

With the availability of more heterogeneous omics data and the development of integrative algorithms that combine such datasets, there is a need for more

complete simulated models of biological cells that model not only transcriptional regulation but model the complete range of interactions between DNA, RNA, proteins and metabolites.

As a first step towards such complete interaction models, the current SynTReN model can be extended to include models of synthetic promoter sequences and to model TF binding with the promoter binding sites. Integrative module and network inference algorithms based on the extended SynTReN model can then use various simulated omics datasets (simulated promoter sequences data, simulated ChIP-chip data and simulated gene expression data) to generate an extensive benchmark of the algorithm's characteristics.

Since the computational power of personal computers doubles about every two years, dynamic simulations of large networks involving multiple biological entities such as genes, proteins and metabolites will become computationally tractable within the next 10-20 years, assuming that Moore's law will hold. Under these circumstances, there is little added value left for static simulation and SynTReN should be converted to a fully dynamic model. The difficulty then may lie not so much in the computational cost of simulations, but rather in the definition of 'good' synthetic models with a set of biologically plausible interactions and in the determination of kinetic parameter ranges that approximate biological reality as closely as possible.

The results in Section 3.8.4 have also indicated that network inference algorithms seem to perform better for network topologies based on biological networks compared to using random graph models. This implies that biological networks have additional topological properties which are currently not covered by random graph models. There are indications in our experiments, which need to be further confirmed, that the information loss in the biological networks is lower than in the random graphs. Interesting research opportunities exist here for designing random graph models that better capture all required topological characteristics of real biological networks. Also the work of Van Leemput [161] concerning the characterization of networks by means of a probabilistic decomposition can provide fruitful ground for designing such random graph models. Little work has also been done on the characterization of heterogeneous regulation networks containing multiple entities such as genes, RNA, proteins, metabolites, etc.

From a network inference perspective, the incorporation of graph topological characteristics of biological networks as prior knowledge for inferring networks may greatly reduce the search space of possible network structures and lead to more robust models of gene regulation. Modeling such information as usable prior knowledge for inference remains still a largely unsolved challenge.

*Pro*Bic

In order to convert *Pro*Bic into a user-friendly software package that is of direct use to a biologist, a number of tasks need to be completed. First, the current version of *Pro*Bic is command line based. While this has proven very useful for running high-throughput experiments, a graphical user interface needs to be developed in order to be of practical use to end users. Second, the query-driven mode of *Pro*Bic is quite sensitive to different choices of priors and their parameters. While a bioinformatician who is 'skilled-in-the-art' can explore this parameter space and define sensible parameter values for a particular dataset, research remains to be done to fully automate this prior and parameter selection. Finally, when the query-driven biclustering is fully automated, it will be most interesting to perform an extensive analysis of the biclustering algorithm on a number of different biological datasets such as the *Escherichia coli*, *Bacillus subtilis* and *Salmonella typhimurium* compendia that are currently being generated at the CMPG.

As proposed in Section 5.8, the *Pro*Bic model is designed to be extended towards identification of cis-regulatory modules. The first extension incorporates promoter sequence data and a second extension describes the integration of microarray condition property data into the *Pro*Bic model. The full power of using PRMs to integrate relational data in combination with an EM strategy is illustrated by combining these two extensions into a single integrative model that identifies cis-regulatory modules with the associated regulatory motifs, experimental conditions and associates these modules with one or more specific condition annotations. These extensions can then be applied to the microarray compendia of the above organisms in combination with the *RegulonDB* database and the condition annotation information that is present in the compendia.

*Pro*Bic is based on the (directed) Probabilistic Relational Model framework. While the framework offers some computational and modeling advantages, certain model extensions may however prove to be more difficult within the context of directed graphical models. More specifically the integration of undirected information such as protein-protein binding information may be more straightforward in either undirected models such as relational Markov networks or in mixed directed and undirected models.

With the advent of huge online databases with interlinked information, an explosion in available data has occurred and especially in the domain of computational biology and many of these datasets are stored in complex multi-relational databases. As described in Chapter 2, several specialized algorithms and models have already been developed to integrate heterogeneous *omics* data to answer specific research questions in e.g. systems biology. However, despite storage and availability of these datasets, the structure, annotation and inter-database relations are far from standardized and uniform. This requires researchers to usually manually perform these integration steps which impedes

for example the direct application of statistical relational learning techniques to such datasets. As uniformity and standardization of these datasets further increases, more automated and large-scale *relational* data mining tasks become within reach of researchers that aim at answering their particular research hypotheses across different knowledge domains, organisms and experimental platforms.

Appendix A

Appendix

A.1 Graph topological measures

Not many deterministic and informative topological measures are available [6]. The established measures can be roughly divided into two categories: high-level (global) measures and low-level (local) measures. In order to calculate the high-level property measures (e.g. average path length) one needs to know the whole network, while the low-level properties can be calculated locally (e.g. marginal degree of individual node).

We use both low and high-level topological measures that address different aspects of the network structure. The high-level measures contain network indices such as average clustering coefficient and average path length. The low-level measures are composed of both marginal and bivariate joint degree distributions [6, 177].

The *average path length* $\bar{l}$ is defined as follows: Given two genes, let l_{ij} be the length of the shortest path connecting these two genes, following the links present in the network. Depending on the type of network, these can either be directed and/or undirected links. N is defined as the number of nodes in the network. The average path length $\bar{l}$ is defined as:

$$\bar{l} = \frac{2}{N(N-1)} \cdot \sum_{i<j}^{N} l_{ij}$$

The *adjacency matrix* ξ_{ij} indicates an interaction between genes γ_i and γ_j ($\xi_{ij} = 1$) or no interaction ($\xi_{ij} = 0$).

The set of nearest neighbors of a gene γ_i is indicated by $\Gamma_i = \{\gamma_j | \xi ij = 1\}$. *The clustering coefficient* C_i for this gene is defined as the ratio between the actual

number of connections between the genes in Γ_i, and the total possible number of connections, $\frac{|\Gamma_i|(|\Gamma_i|-1)}{2}$.

Formally, the clustering coefficient of the i-th gene C_i is defined as:

$$C_i = \frac{2 \cdot L_i}{|\Gamma_i|(|\Gamma_i| - 1)}$$

where

$$L_i = \sum_{j=1}^{N} \xi_{ij} \cdot [\sum_{k \in \Gamma_i} \xi_{jk}]$$

The (average) clustering coefficient is defined as the average C_i over all genes γ_i:

$$C = \frac{1}{N} \sum_{i=1}^{N} C_i$$

A.2 SynTReN performance metrics

Comparison between adjacency matrices derived from the original and inferred networks was performed by counting the corresponding and conflicting entries in both matrices and calculating *sensitivity* (also known as *recall*), *specificity*, and *precision* (also known as *positive predictive value*). As summary metrics *Jaccard index* [148] and *F-measure* can for example be used. An entry is considered to be a *true positive* (TP) or a *true negative* (TN) if it equals 1 or 0, respectively in both matrices. A *false positive* (FP) entry has a value of 1 in the reconstructed network matrix and a value of 0 in the known topology matrix; a *false negative* (FN) entry equals 1 in the known network matrix and 0 in the reconstructed network matrix. The metrics mentioned above can then be calculated as follows:

$$\begin{aligned}
sensitivity = recall &= \frac{TP}{TP + FN} \\
specificity &= \frac{TN}{TN + FP} \\
precision &= \frac{TP}{TP + FP} \\
J = Jaccardindex &= \frac{TP}{FP + FN + TP} \\
F - measure &= \frac{2 \times precision \times recall}{precision + recall}
\end{aligned}$$

All of the performance metrics have a range between 0 and 1, with 1 representing a perfect match and progressively lower scores with increasing number of

false positives or false negatives. As stated in [112], since genetic networks are sparse, potential false positives far exceed potential true positives and specificity by itself might be an inappropriate performance metric as even small deviations from a value of 1 represent quite large numbers of false positives.

Both the Jaccard index and the F-measure can be used as a single measure of performance of the algorithm. The behavior of these metrics was studied by starting from the adjacency matrix of a 50 gene, 76 edge regulatory network and repeatedly adding, removing, or rewiring random a number of edges in the network and comparing the resulting adjacency matrix with the original one. The results of this analysis are shown in Figures A.1a-A.1c and A.1d-A.1f.

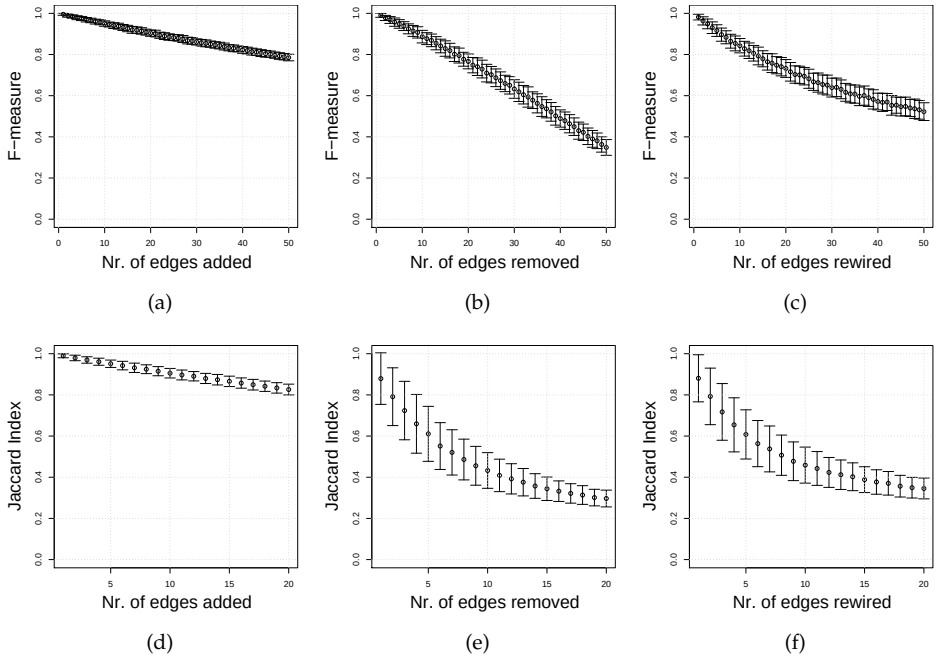


Figure A.1: Evolution of the F-measure and the Jaccard index in function of the number of edge additions, removals or rewirings.

The F-measure shows a near linear decrease with the number of modifications for all three types of change. The Jaccard index declines linearly with edge addition, but shows sublinear behavior for edge removal or edge rewiring. Moreover the spread in values of the F-measure is substantially smaller compared to that of the Jaccard index. Given the fact that the F-measure shows smoother degradation it is preferred over Jaccard index in the results shown later. We conclude that these metrics are different in nature, in that the F-

measure exhibits gentle degradation in value when generally moving away from the original network, while the Jaccard index lacks this behavior. We have therefore adopted the F-measure as standard summarizing metric in the experiments.

A.3 Conjugate prior distributions for *Pro*Bic model

Different conjugate priors can be defined for the *Pro*Bic model $(\mu_{a,b}, \sigma_{a,b})$ distributions from Section 5.4.5. For example, any member of the exponential family is a conjugate prior to Equation 4.13. We will highlight two members of the exponential family here, namely the Normal distribution and the Normal-Inverse-χ^2 distribution.

A.3.1 Normal distribution prior

A straightforward choice for the prior distribution is a Normal distribution:

$$P(\mu_{a,b}, \sigma_{a,b}) = \frac{1}{\sigma_{a,b}\sqrt{2\pi}} \exp[-\frac{ssq_p - 2\mu_{a,b}s_p + n_p\mu_{a,b}^2}{2\sigma_{a,b}^2}] \tag{A.1}$$

where the set of sufficient statistics (n_0, s_0, ssq_0) parameterize the prior distribution. These sufficient statistics can be considered as *pseudo values*, since the prior behaves as if these additional values were added to the existing set of expression values of the likelihood distribution $P(X)$.

- n_0: the number of pseudo expression values
- s_0: sum of the pseudo expression values
- ssq_0: sum of the squared pseudo expression values

The posterior distribution $P(\mu_{a,b}, \sigma_{a,b}) \cdot P(X)$, where X represents the (expression) data, is again a Normal distribution characterized by:

$$n_p = n_0 + n_X \tag{A.2}$$
$$s_p = s_0 + s_X \tag{A.3}$$
$$ssq_p = ssq_0 + ssq_X \tag{A.4}$$

A.3.2 Normal-Inverse-χ^2 distribution prior

The Normal-Inverse-χ^2 distribution is a very useful but slightly more complicated prior. It defines a prior distribution directly for the mean and standard

deviation in the following way:

$$\sigma^2 \qquad Inv - \chi^2(\nu_0, \sigma_0^2) \tag{A.5}$$

$$\mu|\sigma^2 \qquad N(\mu_0, \sigma^2/\kappa_0) \tag{A.6}$$

This leads to the following formal definition of the prior which is parameterized by $(\mu_0, \kappa_0, \nu_0, \sigma_0^2)$:

$$P(\mu_{a,b}, \sigma_{a,b}) = \sigma^{-1}(\sigma^2)^{-\nu_0/2+1} exp\left(-\frac{1}{2\sigma^2}[\nu_0\sigma_0^2 + \kappa_0(\mu_0 - \mu)^2]\right) \tag{A.7}$$

The posterior distribution $P(\mu_{a,b}, \sigma_{a,b}) \cdot P(X)$, where X represents the (expression) data, is again a Normal-Inverse-χ^2 distribution characterized by:

$$\mu_p = \frac{\kappa_0}{\kappa_0 + n_X}\mu_0 + \frac{\kappa_0}{\kappa_0 + n_X}\frac{s_X}{n_X} \tag{A.8}$$

$$\kappa_p = \kappa_0 + n_X \tag{A.9}$$

$$\nu_p = \nu_0 + n_X \tag{A.10}$$

$$\nu_p\sigma_p^2 = \nu_0\sigma_0^2 + (n_X - 1)ssq + \frac{\kappa_0 n_X}{\kappa_0 + n_X}(\frac{s_X}{n_X} - \mu_0)^2 \tag{A.11}$$

The posterior mean and variance for σ^2 are:

$$E(\sigma^2) = \frac{\nu_p}{\nu_p - 2}\sigma_p^2 \tag{A.12}$$

$$var(\sigma^2) = \frac{2\nu_p^2}{(\nu_p - 2)^2(\nu_p - 4)}\sigma_p^4 \tag{A.13}$$

$$\tag{A.14}$$

and for the mean μ given σ^2:

$$E(\mu|\sigma^2) = \mu_p \tag{A.15}$$

$$var(\mu|\sigma^2) = \sigma^2/\kappa_p \tag{A.16}$$

$$\tag{A.17}$$

A.4 Necessary conditions for the E-step optimization

As described in Section 5.5.2, one could hypothesize that assigning a gene(array) to two biclusters X and Y will not lead to an improvement in score compared to

each of the single bicluster assignments, if the score of assigning the gene(array) to a single bicluster X or Y would not lead to a score improvement compared to the background assignment.

If this hypothesis holds true, an Apriori [2] approach can be used to evaluate only $a.B$ vectors with potentially good scores by applying the following procedure. In the first iteration (level 0), the score for the background assignment $a.B = []$ is calculated. For all next levels, the following steps are iterated until no candidate assignments remain:

- at level L, a list of candidate assignments $a.B$ with L biclusters is constructed where an assignment $a.B$ is allowed only if all the $(L-1)$ subvectors of $a.B$ were a candidate assignment in the previous level $(L-1)$. A $(L-1)$ subvector of $a.B$ contains $(L-1)$ biclusters that are all a bicluster of $a.B$.
- The score is calculated for this candidate assignment according to Equation 5.34 and the candidate assignment is kept only if this score is higher than each of the $(L-1)$ subvectors.

Finally the $a.B$ assignment with the highest score is selected as the maximum likelihood solution of Equation 5.34.

We will now formalize our hypothesis and derive the necessary conditions for this hypothesis to hold true in case of overlap between two biclusters:

$$\begin{aligned} &\textbf{if } score([1]) < score([]) \textbf{ or } score([2]) < score([]) \\ &\textbf{then } score([1,2]) < score([1]) \textbf{ or } score([1,2]) < score([2]) \end{aligned} \tag{A.18}$$

From Equation 5.34, we can derive the scores for each of the possible $a.B$ assignments:

$$score(a.B = []) = v_{a,[1],[]} + v_{a,[1,2],[]} + v_{a,[2],[]} + v_{a,[],[]} \tag{A.19}$$

$$score(a.B = [1]) = v_{a,[1],[1]} + v_{a,[1,2],[1]} + v_{a,[2],[]} + v_{a,[],[]} \tag{A.20}$$

$$score(a.B = [2]) = v_{a,[1],[]} + v_{a,[1,2],[2]} + v_{a,[2],[2]} + v_{a,[],[]} \tag{A.21}$$

$$score(a.B = [1,2]) = v_{a,[1],[1]} + \frac{v_{a,[1,2],[1]} + v_{a,[1,2],[2]}}{2} + v_{a,[2],[2]} + v_{a,[],[]} \tag{A.22}$$

This leads to the following $\delta(score)$ expressions:

$$score(a.B = [1]) - score(a.B = []) = \delta_{a,[1],[1]} + \delta_{a,[1,2],[1]} \tag{A.23}$$

$$score(a.B = [2]) - score(a.B = []) = \delta_{a,[1,2],[2]} + \delta_{a,[2],[2]} \tag{A.24}$$

$$score(a.B = [1,2]) - score(a.B = [1]) = \frac{\delta_{a,[1,2],[2]} - \delta_{a,[1,2],[1]}}{2} + \delta_{a,[2],[2]} \tag{A.25}$$

$$score(a.B = [1,2]) - score(a.B = [2]) = \frac{\delta_{a,[1,2],[1]} - \delta_{a,[1,2],[2]}}{2} + \delta_{a,[1],[1]} \tag{A.26}$$

with:

$$\delta_{a,[1],[1]} = v_{a,[1],[1]} - v_{a,[1],[]} \quad (A.27)$$

$$\delta_{a,[1,2],[1]} = v_{a,[1,2],[1]} - v_{a,[1,2],[]} \quad (A.28)$$

$$\delta_{a,[1,2],[2]} = v_{a,[1,2],[2]} - v_{a,[1,2],[]} \quad (A.29)$$

$$\delta_{a,[2],[2]} = v_{a,[2],[2]} - v_{a,[2],[]} \quad (A.30)$$

Hypothesis A.18 can be split in two cases: (a) $score([1]) < score([])$ and (b) $score([2]) < score([])$ (the case where both (a) and (b) are fulfilled, is implicitly covered). We now derive the necessary conditions for case (a), the conditions for case (b) can be derived analogously.

From (a), we derive:

$$score([1] < score([]) \Leftrightarrow \delta_{a,[1],[1]} + \delta_{a,[1,2],[1]} < 0 \quad (A.31)$$

leading to the necessary conditions:

$$score(a.B = [1,2]) - score(a.B = [1]) < 0 \Leftrightarrow \frac{\delta_{a,[1,2],[2]} - \delta_{a,[1,2],[1]}}{2} < \delta_{a,[2],[2]} \quad (A.32)$$

These necessary conditions are met for example if the overlap region between two biclusters is small. In that case the score change $\delta_{a,[1,2],[2]}$ for adding the overlap genes from the background to bicluster 2 and the score change $\delta_{a,[1,2],[1]}$ for adding the overlap genes from the background to bicluster 1, are both small compared to the score change $\delta_{a,[2],[2]}$ for adding all non-overlapping genes of bicluster 2 from the background to bicluster 2, so Equation A.32 holds.

Bibliography

[1] Affymetrix. Microarray suite user guide. http://www.affymetrix.com/support/technical/index.affx.

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C., 1993.

[3] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[4] T. Akutsu, S. Miyano, and S. Kuhara. Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. *Pacific Symposium on Biocomputing*, pages 17–28, 1999.

[5] R. Albert and A. Barabási. Topology of evolving networks: local events and universality. *Physical Review Letters*, 85:5234–5237, December 2000.

[6] R. Albert and A. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–97, 2002.

[7] O. T. Avery, C. M. MacLeod, and M. McCarty. Studies on the chemical nature of the substance inducing transformation of pneumococcal types. Inductions of transformation by a desoxyribonucleic acid fraction isolated from *Pneumococcus* type III. *The Journal of Experimental Medicine*, 149(2):297–326, February 1979.

[8] BA Education. The people responsible for the discovery of DNA. url: http://www.ba-education.demon.co.uk/for/science/dnamain.html, 2007.

[9] M. M. Babu, N. M. Luscombe, L. Aravind, M. Gerstein, and S. A. Teichmann. Structure and evolution of transcriptional regulatory networks. *Current Opinion in Structural Biology*, 14(3):283–291, June 2004.

[10] P. Baldi and A. D. Long. A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes. *Bioinformatics*, 17(6):509–519, June 2001.

[11] M. Bansal, G. D. Gatta, and D. di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, April 2006.

[12] Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21(11):1337–1342, November 2003.

[13] T. Barrett, D. B. Troup, S. E. Wilhite, P. Ledoux, D. Rudnev, C. Evangelista, I. F. Kim, A. Soboleva, M. Tomashevsky, and R. Edgar. NCBI GEO: mining tens of millions of expression profiles–database and tools update. *Nucleic Acids Research*, 35(Database issue), January 2007.

[14] K. Basso, A. A. Margolin, G. Stolovitzky, U. Klein, D. R. Favera, and A. Califano. Reverse engineering of regulatory networks in human B cells. *Nature Genetics*, 37:382–390, April 2005.

[15] W. Bateson. *Mendel's Principles of Heredity, a Defense.* Cambridge University Press, London, 1902.

[16] A. Battle, E. Segal, and D. Koller. Probabilistic discovery of overlapping cellular processes and their regulation. *Journal of Computational Biology*, 12(7):909–927, September 2005.

[17] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. *Journal of Computational Biology*, 10(3-4):373–384, 2003.

[18] S. A. Benner and M. A. Sismour. Synthetic biology. *Nature Reviews Genetics*, 6(7):533–543, July 2005.

[19] S. Bergmann, J. Ihmels, and N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical Review E*, 67(3):031902+, March 2003.

[20] A. Bernard and A. J. Hartemink. Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data. *Pacific Symposium on Biocomputing*, pages 459–470, 2005.

[21] A. P. Bird. CpG-rich islands and the function of DNA methylation. *Nature*, 321(6067):209–213, May 1986.

[22] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, November 1995.

[23] B. Bollobás. *Random graphs*. Academic Press, New York, 1985.

[24] B. Bollobás, C. Borgs, C. Chayes, and O. Riordan. Directed scale-free graphs. *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 132–139, 2003.

[25] S. Borman. The Expectation Maximization algorithm: A short tutorial. 2006.

[26] A. Brazma and J. Vilo. Gene expression data analysis. *FEBS letters*, 480(1):17–24, August 2000.

[27] H. J. Bussemaker, H. Li, and E. D. Siggia. Regulatory element detection using correlation with expression. *Nature Genetics*, 27(2):167–171, February 2001.

[28] G. Cagney, D. Alvaro, R. J. D. Reid, P. H. Thorpe, R. Rothstein, and N. J. Krogan. Functional genomics of the yeast dna-damage response. *Genome Biology*, 7:233+, September 2006.

[29] Y. Cheng and G. M. Church. Biclustering of expression data. *Proceedings of the 8th Annual International Conference Intelligent Systems for Molecular Biology*, 8:93–103, 2000.

[30] D. M. Chickering, D. Geiger, and D. Heckerman. Learning bayesian networks is NP-hard. Technical Report MSR-TR-94-17, Microsoft research, November 1994.

[31] J.-M. Claverie. Fewer genes, more noncoding RNA. *Science*, 309(5740):1529–1530, September 2005.

[32] G. F. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 09(4):309–347, October 1992.

[33] F. H. C. Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[34] B. D'Ambrosio, J. Jorgensen, and E. Altendorf. Ecosystem analysis using probabilistic relational modeling. In *Proceedings of the IJCAI-2003 Workshop on Learning Statistical Models from Relational Data*, Acapulco, Mexico, 2003.

[35] C. Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. John Murray, London, 1st edition, 1859.

[36] S. Datta and S. Datta. Comparisons and validation of statistical clustering techniques for microarray gene expression data. *Bioinformatics*, 19(4):459–466, March 2003.

[37] T. De Bie, P. Monsieurs, K. Engelen, B. De Moor, N. Cristianini, and K. Marchal. Discovering regulatory modules from heterogeneous information sources. In *Pacific Symposium on Biocomputing*, 2005.

[38] H. de Jong. Modeling and simulation of genetic regulatory systems: a literature review. *Journal of Computational Biology*, 9(1):67–103, 2002.

[39] L. De Raedt, editor. *Logical and Relational Learning*. Springer, 2008.

[40] L. De Raedt and K. Kersting. Probabilistic logic learning. Technical report, 2004.

[41] L. De Raedt and K. Kersting. Probabilistic Logic Learning. *ACM-SIGKDD Explorations: Special issue on Multi-Relational Data Mining*, 5:2003, 2004.

[42] J. Demeter, C. Beauheim, J. Gollub, T. Hernandez-Boussard, H. Jin, D. Maier, J. C. Matese, M. Nitzberg, F. Wymore, Z. K. Zachariah, P. O. Brown, G. Sherlock, and C. A. Ball. The Stanford Microarray Database: implementation of new analysis tools and open source release of software. *Nucleic Acids Research*, 35(Database issue), January 2007.

[43] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, 1977.

[44] I. Derényi, I. Farkas, G. Palla, and T. Vicsek. Topological phase transitions of random networks. *Physica A*, 334:583–590, 2004.

[45] P. D'Haeseleer, X. Wen, S. Fuhrman, and R. Somogyi. Linear modeling of mRNA expression levels during CNS development and injury. *Pacific Symposium on Biocomputing*, pages 41–52, 1999.

[46] I. S. Dhillon, E. M. Marcotte, and U. Roshan. Diametrical clustering for identifying anti-correlated gene clusters. *Bioinformatics*, 19(13):1612–1619, 2003.

[47] T. Dhollander, Q. Sheng, K. Lemmens, B. De Moor, K. Marchal, and Y. Moreau. Query-driven module discovery in microarray data. *Bioinformatics*, 23(19):2573–2580, October 2007.

[48] N. Dojer, A. Gambin, A. Mizera, B. Wilczyński, and J. Tiuryn. Applying dynamic Bayesian networks to perturbed gene expression data. *BMC Bioinformatics*, 7, 2006.

[49] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, December 1998.

[50] M. B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, January 2000.

[51] M. B. Elowitz, A. J. Levine, E. D. Siggia, and P. S. Swain. Stochastic gene expression in a single cell. *Science*, 297(5584):1183–1186, August 2002.

[52] P. Erdös and A. Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[53] A. Fersht. *Enzyme structure and mechanism*, volume 2. W.H. Freeman and Company, New York, 1985.

[54] N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science*, 303:799–805, February 2004.

[55] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

[56] F. Gao, B. C. Foat, and H. J. Bussemaker. Defining transcriptional networks through integrative modeling of mRNA expression and transcription factor binding data. *BMC Bioinformatics*, 5, March 2004.

[57] T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in *Escherichia coli*. *Nature*, 403(6767):339–342, January 2000.

[58] A. P. Gasch and M. B. Eisen. Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome Biology*, 3(11), October 2002.

[59] A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11(12):4241–4257, December 2000.

[60] L. Getoor. *Introduction to Statistical Relational Learning*. The MIT Press, August 2007.

[61] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure. In *Proceedings of the 18th International Conference on Machine Learning*, pages 170–177. Morgan Kaufmann, San Francisco, CA, 2001.

[62] L. Getoor, E. Segal, B. Taskar, and D. Koller. Probabilistic models of text and link structure for hypertext classification. In *IJCAI01 Workshop on Text Learning: Beyond Supervision*, 2001.

[63] G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. *Proceedings of the National Academy of Sciences*, 97(22):12079–12084, October 2000.

[64] J. Gu and J. Liu. Bayesian biclustering of gene expression data. *BMC Genomics*, 9(Suppl 1), 2008.

[65] N. Guelzim, S. Bottani, P. Bourgine, and F. Kepes. Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics*, 31:60–63, May 2002.

[66] C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, K. D. Macisaac, T. W. Danford, N. M. Hannett, J. B. Tagne, D. B. Reynolds, J. Yoo, E. G. Jennings, J. Zeitlinger, D. K. Pokholok, M. Kellis, P. A. Rolfe, K. T. Takusagawa, E. S. Lander, D. K. Gifford, E. Fraenkel, and R. A. Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431:99–104, September 2004.

[67] D. Hart and E. Jones. *Genetics: Analysis of Genes and Genomes*. Jones and Bartlett Publishers, 5th edition, 2001.

[68] A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and R. A. Young. Combining location and expression data for principled discovery of genetic regulatory network models. *Pacific Symposium on Biocomputing*, 2002.

[69] A. J. Hartemink and E. Segal. Joint learning from multiple types of genomic data. *Pacific Symposium on Biocomputing*, pages 445–446, 2005.

[70] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402(6761 Suppl), December 1999.

[71] D. Heckerman and D. Geiger. Learning Bayesian Networks. Technical Report MSR-TR-95-02, Microsoft Research, Redmond, WA, December 1994.

[72] M. J. Herrgård, M. W. Covert, and B. Palsson. Reconciling gene expression data with known genome-scale regulatory network structures. *Genome Research*, 13(11):2423–2434, November 2003.

[73] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 688–693, 1999.

[74] J. H. Hofmeyr and C. A. Bowden. The reversible Hill equation: how to incorporate cooperative enzymes into metabolic models. *Computer Applications In The Biosciences*, 13:377–385, August 1997.

[75] T. R. Hughes, M. Mao, A. R. Jones, J. Burchard, M. J. Marton, K. W. Shannon, S. M. Lefkowitz, M. Ziman, J. M. Schelter, M. R. Meyer, S. Kobayashi, C. Davis, H. Dai, Y. D. He, S. B. Stephaniants, G. Cavet, W. L. Walker, A. West, E. Coffey, D. D. Shoemaker, R. Stoughton, A. P. Blanchard, S. H. Friend, and P. S. Linsley. Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer. *Nature Biotechnology*, 19(4):342–347, April 2001.

[76] T. R. Hughes, M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, C. D. Armour, H. A. Bennett, E. Coffey, H. Dai, Y. D. He, M. J. Kidd, A. M. King, M. R. Meyer, D. Slade, P. Y. Lum, S. B. Stepaniants, D. D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. H. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–26, 2000.

[77] D. Husmeier. Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics*, 19:2271–2282, November 2003.

[78] J. Ihmels, S. Bergmann, and N. Barkai. Defining transcription modules using large-scale gene expression data. *Bioinformatics*, 20(13):1993–2003, September 2004.

[79] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genetics*, 31:370–377, 2002.

[80] S. Imoto, T. Higuchi, T. Goto, K. Tashiro, S. Kuhara, and S. Miyano. Combining microarrays and biological knowledge for estimating gene networks via bayesian networks. *Journal of Bioinformatics and Computational Biology*, 2(1):77–98, March 2004.

[81] International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome. *Nature*, 431(7011):931–945, October 2004.

[82] R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis, U. Scherf, and T. P. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, April 2003.

[83] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Information Science and Statistics. Springer, July 2001.

[84] M. Jordan. *Learning in Graphical Models*. The MIT Press, November 1998.

[85] A. Joshi, R. De Smet, K. Marchal, Y. Van de Peer, and T. Michoel. Module networks revisited: computational assessment and prioritization of model predictions. *Bioinformatics*, 25(4):490–496, February 2009.

[86] M. Kato, N. Hata, N. Banerjee, B. Futcher, and M. Q. Zhang. Identifying combinatorial regulation of transcription factors and binding motifs. *Genome Biology*, 5(8):R56+, July 2004.

[87] M. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence*, pages 282–29, San Francisco, CA, 1997. Morgan Kaufmann.

[88] K. Kersting and L. De Raedt. Bayesian Logic Programs. Technical report, Nov 2001.

[89] H. Kim, W. Hu, and Y. Kluger. Unraveling condition specific gene transcriptional regulatory networks in *Saccharomyces cerevisiae*. *BMC Bioinformatics*, 7(1):165+, 2006.

[90] M. Kloster, C. Tang, and N. S. Wingreen. Finding regulatory modules through large-scale gene-expression data analysis. *Bioinformatics*, 21(7):1172–1179, April 2005.

[91] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein. Spectral biclustering of microarray data: coclustering genes and conditions. *Genome Research*, 13(4):703–716, April 2003.

[92] C. Knüpfer, P. Dittrich, and C. Beckstein. *Artificial Gene Regulation: A Data Source for Validation of Reverse Bioengineering*, pages 66–75. Akademische Verlagsgesellschaft Aka, Berlin., April 2004.

[93] D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 302–313, 1997.

[94] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the Fifteenth Conference of the American Association for Artificial Intelligence*, pages 580–587, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[95] D. A. Kulesh, D. R. Clive, D. S. Zarlenga, and J. J. Greene. Identification of interferon-modulated proliferation-related cDNA sequences. *Proceedings of the National Academy of Sciences*, 84(23):8453–8457, December 1987.

[96] R. Laughlin. *A Different Universe: Reinventing Physics from the Bottom Down*. Basic Books, 2005.

[97] Y. Lazebnik. Can a biologist fix a radio? – Or, what I learned while studying apoptosis. *Cancer Cell*, 2(3):179–182, September 2002.

[98] L. Lazzeroni and A. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12:61–86, January 1999.

[99] P. H. Lee and D. Lee. Modularized learning of genetic interaction networks from biological annotations and mRNA expression data. *Bioinformatics*, 21(11):2739–2747, June 2005.

[100] R. C. Lee, R. L. Feinbaum, and V. Ambros. The *C. elegans* heterochronic gene *lin-4* encodes small RNAs with antisense complementarity to *lin-14*. *Cell*, 75(5):843–854, December 1993.

[101] T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. B. Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, B. D. Gordon, B. Ren, J. J. Wyrick, J. B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 298(5594):799–804, Oct 2002.

[102] K. Lemmens, T. De Bie, T. Dhollander, S. De Keersmaecker, I. Thijs, G. Schoofs, A. De Weerdt, B. De Moor, J. Vanderleyden, J. Collado-Vides, K. Engelen, and K. Marchal. DISTILLER: a data integration framework to reveal condition dependency of complex regulons in *Escherichia coli*. *Genome Biology*, 10:R27+, March 2009.

[103] Y. F. Leung and D. Cavalieri. Fundamentals of cDNA microarray data analysis. *Trends in Genetics*, 19(11):649–659, November 2003.

[104] E. Li, C. Beard, and R. Jaenisch. Role for DNA methylation in genomic imprinting. *Nature*, 366(6453):362–365, November 1993.

[105] H. Li and W. Wang. Dissecting the transcription networks of a cell using computational genomics. *Current opinion in genetics & development*, 13(6):611–616, December 2003.

[106] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[107] Y. Luan and H. Li. Clustering of time-course gene expression data using a mixed-effects model with b-splines. *Bioinformatics*, 19(4):474–482, March 2003.

[108] H. W. Ma, B. Kumar, U. Ditges, F. Gunzer, J. Buer, and A. P. Zeng. An extended transcriptional regulatory network of *Escherichia coli* and analysis of its hierarchical structure and network motifs. *Nucleic Acids Research*, 32:6643–6649, 2004.

[109] Madan, S. A. Teichmann, and L. Aravind. Evolutionary dynamics of prokaryotic transcriptional regulatory networks. *Journal of Molecular Biology*, 358(2):614–633, April 2006.

[110] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[111] S. C. Madeira and A. L. Oliveira. A linear time biclustering algorithm for time series expression data. In *Proceedings of the 5th Workshop on Algorithms in Bioinformatics*, pages 39–52, 2005.

[112] A. A. Margolin, I. Nemenman, K. Basso, U. Klein, C. Wiggins, G. Stolovitzky, R. D. Favera, and A. Califano. ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7(Suppl 1), 2006.

[113] G. J. Mclachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. Wiley-Interscience, October 2007.

[114] P. Mendes, W. Sha, and K. Ye. Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, 19:II122–II129, October 2003.

[115] E. Merzbacher. *Quantum Mechanics*. John Wiley, New York, second edition, December 1970.

[116] T. Michoel, S. Maere, E. Bonnet, A. Joshi, Y. Saeys, T. Van den Bulcke, K. Van Leemput, P. van Remortel, M. Kuiper, K. Marchal, and Y. Van de Peer. Validating module network learning algorithms using simulated data. *BMC Bioinformatics*, 8(Suppl 2), 2007.

[117] F. Miescher. Ueber die chemische Zusammensetzung der Eiterzellen. *Med.-Chem. Unters*, 4:441–460, 1871.

[118] R. Milo, S. S. Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298:824–827, October 2002.

[119] I. Nachman, A. Regev, and N. Friedman. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics*, 20:I248–I256, August 2004.

[120] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, April 2003.

[121] J. Neville and D. Jensen. Iterative classification in relational data. Technical report, 2000.

[122] J. Newton and R. Greiner. Hierarchical probabilistic relational models for collaborative filtering. In *Proceedings of the Workshop on Statistical Relational Learning, 21st International Conference on Machine Learning*, pages 249–263, 2004.

[123] A. B. Owen, J. Stuart, K. Mach, A. M. Villeneuve, and S. Kim. A gene recommender algorithm to identify coexpressed genes in *C. elegans*. *Genome Research*, 13(8):1828–1837, August 2003.

[124] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, second edition, 1984.

[125] H. Parkinson, M. Kapushesky, M. Shojatalab, N. Abeygunawardena, R. Coulson, A. Farne, E. Holloway, N. Kolesnykov, P. Lilja, M. Lukk, R. Mani, T. Rayner, A. Sharma, E. William, U. Sarkans, and A. Brazma. ArrayExpress – a public database of microarray experiments and gene expression profiles. *Nucleic Acids Research*, 35(Database issue), January 2007.

[126] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, September 1988.

[127] D. Pe'er, A. Regev, G. Elidan, and N. Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17:S215–S224, 2001.

[128] A. Pfeffer, D. Koller, B. Milch, and K. Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Annual Conference on Uncertainty in AI*, pages 541–550, 1999.

[129] D. Pinkel and D. G. Albertson. Array comparative genomic hybridization and its applications in cancer. *Nature Genetics*, 37 Suppl, June 2005.

[130] G. D. Plotkin. *Machine Intelligence*, pages 153–163. Edinburgh University Press, 1970.

[131] A. Prelic, S. Bleuler, P. Zimmermann, A. Wille, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, May 2006.

[132] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986.

[133] C. V. Rao, D. M. Wolf, and A. P. Arkin. Control, exploitation and tolerance of intracellular noise. *Nature*, 420(6912):231–237, November 2002.

[134] T. Reil. Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In *European Conference on Artificial Life*, pages 457–466, 1999.

[135] D. J. Reiss, N. S. Baliga, and R. Bonneau. Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. *BMC Bioinformatics*, 7:280+, June 2006.

[136] B. Ren, F. Robert, J. J. Wyrick, O. Aparicio, E. G. Jennings, I. Simon, J. Zeitlinger, J. Schreiber, N. Hannett, E. Kanin, T. L. Volkert, C. J. Wilson, S. P. Bell, and R. A. Young. Genome-wide location and function of dna binding proteins. *Science*, 290(5500):2306–2309, December 2000.

[137] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, February 2006.

[138] D. M. Rocke and B. Durbin. A model for measurement error for gene expression arrays. *Journal of Computational Biology*, 8:557–569, 2001.

[139] S. Rogers and M. Girolami. A bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics*, 21(14):3131–3137, July 2005.

[140] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270(5235):467–470, October 1995.

[141] A. Schumacher, P. Kapranov, Z. Kaminsky, J. Flanagan, A. Assadzadeh, P. Yau, C. Virtanen, N. Winegarden, J. Cheng, T. Gingeras, and A. Petronis. Microarray-based DNA methylation profiling: technology and applications. *Nucleic Acids Research*, 34(2):528–542, 2006.

[142] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[143] E. Segal. *Rich probabilistic models for genomic data*. PhD thesis, Stanford University, 2004.

[144] E. Segal, A. Battle, and D. Koller. Decomposing gene expression into cellular processes. *Pacific Symposium on Biocomputing*, pages 89–100, 2003.

[145] E. Segal, M. Shapira, A. Regev, D. Pe'er, D. Botstein, D. Koller, and N. Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature Genetics*, 34(2):166–176, June 2003.

[146] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller. Rich probabilistic models for gene expression. *Bioinformatics*, 17:S243–S252, 2001.

[147] E. Segal, R. Yelensky, and D. Koller. Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. *Bioinformatics*, 19 Suppl 1:I273–I282, 2003.

[148] B. E. Shakhnovich, T. E. Reddy, K. Galinsky, J. Mellor, and C. Delisi. Comparisons of predicted genetic modules: identification of co-expressed genes through module gene flow. *Genome Informatics*, 15(1):221–8, 2004.

[149] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nature Genetics*, 31:64–68, May 2002.

[150] Q. Sheng, Y. Moreau, and B. De Moor. Biclustering microarray data by Gibbs sampling. *Bioinformatics*, 19 Suppl 2, October 2003.

[151] V. A. Smith, E. D. Jarvis, and A. J. Hartemink. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, 18:S216–S224, 2002.

[152] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297, December 1998.

[153] Y. Tamada, S. Kim, H. Bannai, S. Imoto, K. Tashiro, S. Kuhara, and S. Miyano. Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection. *Bioinformatics*, 19:II227–II236, October 2003.

[154] A. Tanay, R. Sharan, M. Kupiec, and R. Shamir. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data. *Proceedings of the National Academy of Sciences*, 101(9):2981–2986, March 2004.

[155] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18 Suppl 1, 2002.

[156] A. Tanay, I. Steinfeld, M. Kupiec, and R. Shamir. Integrative analysis of genome-wide experiments in the context of a large high-throughput data compendium. *Molecular Systems Biology*, 1(1):E1–E10, March 2005.

[157] J. Tegnér and J. Björkegren. Perturbations to uncover gene networks. *Trends in Genetics*, November 2006.

[158] L. H. Ungar and D. P. Foster. A formal statistical approach to collaborative filtering. In *In CONALD'98*, 1998.

[159] T. Van den Bulcke, K. Lemmens, Y. Van de Peer, and K. Marchal. Inferring transcriptional networks by mining 'omics' data. *Current bioinformatics*, 1, 2006.

[160] T. Van den Bulcke, K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor, and K. Marchal. SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7, 2006.

[161] K. Van Leemput. *Evaluation of network inference algorithms by means of synthetic data and characterization of graphs using probabilistic decomposition*. PhD thesis, Universiteit Antwerpen, 2008.

[162] K. Van Leemput, T. Van den Bulcke, T. Dhollander, B. De Moor, K. Marchal, and P. van Remortel. Exploring the operational characteristics of inference algorithms for transcriptional networks by means of synthetic data. *Artificial Life*, 14(1):49–63, 2008.

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

[164] W. Wang, M. J. Cherry, D. Botstein, and H. Li. A systematic approach to reconstructing transcription networks in *Saccharomyces cerevisiae*. *Proceedings of the National Academy of Sciences*, 99(26):16893–16898, December 2002.

[165] J. D. Watson and F. H. Crick. Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737–738, April 1953.

[166] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

[167] C. Wei, J. Li, and R. E. Bumgarner. Sample size for detecting differentially expressed genes in microarray experiments. *BMC Genomics*, 5(1), November 2004.

[168] T. Wolf, B. Brors, T. Hofmann, and E. Georgii. Global biclustering of microarray data. In S. Tsumoto, C. W. Clifton, N. Zhong, X. Wu, J. Liu, B. W. Wah, and Y.-M. Cheung, editors, *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops (ICDMW 2006)*, pages 125–129, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[169] C. J. Wu and S. Kasif. Gems: a web server for biclustering analysis of expression data. *Nucleic Acids Research*, 33(Web Server issue), July 2005.

[170] J. Wu, L. T. Smith, C. Plass, and T. H. Huang. Chip-chip comes of age for genome-wide functional analysis. *Cancer Research*, 66(14):6899–6902, July 2006.

[171] J. F. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[172] X. Xu, L. Wang, and D. Ding. Learning module networks from genome-wide location and expression data. *FEBS Letters*, 578:297–304, December 2004.

[173] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *Proceedings of 3rd IEEE Symposium on BioInformatics and BioEngineering*, pages 321–327, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[174] Y. H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed. Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Research*, 30, February 2002.

[175] K. Y. Yeung and W. L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, September 2001.

[176] D. E. Zak, F. J. Doyle, and J. S. Schwaber. Simulation studies for the identification of genetic networks from cDNA array and regulatory activity data. *Proceedings of the Second International Conference on Systems Biology*, pages 231–238, 2001.

[177] D. Zhu and Z. S. Qin. Structural comparison of metabolic networks in selected single cell organisms. *BMC Bioinformatics*, 6, January 2005.

Curriculum Vitae

Personal data

name:	Tim Van den Bulcke
date and place of birth:	July 3, 1977 in Ghent (Belgium)
address:	Gentsesteenweg 45 2800 Mechelen (Belgium)
email:	tvandenbulcke@gmail.com

Education

1989 - 1995	Latin-Mathematics, Sint-Maarteninstituut (Aalst).
1995 - 2000	Civil engineer in physics, UGent (Gent). Magna cum laude thesis "Study and implementation of a Field Programmable Neural Array"
2000 - 2001	Master of Artificial Intelligence, KULeuven (Leuven). Option: engineering and computer science Cum laude Project "An artificial intelligent GO player: static Semeai evaluation"

Research and work experience

2001 - 2004	Tibotec, Johnson&Johnson (Mechelen). Scientist at the Chem- & Bio-Informatics department
2004 - present	ESAT-SCD, KULeuven (Leuven). PhD "Robust algorithms for inferring regulatory networks, based on expression measurements and biological prior information."

Publication list

Journal papers

Published

- T. Van den Bulcke, K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor and K. Marchal. *SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms*. BMC Bioinformatics 2006, 7:43.

- T. Van den Bulcke, K. Lemmens, Y. Van de Peer, and K. Marchal. *Inferring transcriptional networks by mining 'omics' data*. Current Bioinformatics 2006, 1:3.

- T. Michoel, S. Maere, E. Bonnet, A. Joshi, T. Van den Bulcke, K. Van Leemput, P. van Remortel, M. Kuiper, K. Marchal and Y. Van de Peer. *Validating module network learning algorithms using simulated data*. BMC Bioinformatics 2007, 8(Suppl 2):S5.

- K. Van Leemput, T. Van den Bulcke, T. Dhollander, B. De Moor, K. Marchal, P. van Remortel. *Exploring the operational characteristics of inference algorithms for transcriptional networks by means of synthetic data*. Artificial Life 2008, 14:1, 49-63.

Submitted

- H. Sung, K. Lemmens, T. Van den Bulcke, K. Engelen, B. De Moor, K. Marchal. *ViTraM: Visualization of Transcriptional Modules.* Bioinformatics 2009.

In preparation

- T. Van den Bulcke, H. Zhao, K. Engelen, B. De Moor and K. Marchal. *Combining global and query-driven biclustering of gene expression data using Probabilistic Relational Models.*

Oral presentations

- T. Van den Bulcke, K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor and K. Marchal. *Analyzing structure learning algorithms by means of synthetic gene expression data*. Seminar, University of East Anglia, Norwich (UK), 2007.
- T. Van den Bulcke, H. Zhao, K. Engelen, B. De Moor, K. Marchal. *ProBic: identification of overlapping biclusters using Probabilistic Relational Models, applied to simulated gene expression data*. Presented at Probabilistic Modelling of Networks and Pathways workshop, Sheffield (UK), 2007.
- T. Van den Bulcke, H. Zhao, K. Engelen, B. De Moor, K. Marchal. *ProBic: identification of overlapping biclusters using Probabilistic Relational Models*. Presented at Probabilistic Modelling in Computational Biology satellite workshop, ISMB/ECCB, Vienna (Austria), 2007.