Erzeugung von positiv definiten Matrizen mit Nebenbedingungen zur Validierung von Netzwerkalgorithmen für Microarray-Daten

Markus Ruschhaupt

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

München, den 22. Jannuar 2008

Erzeugung von positiv definiten Matrizen mit Nebenbedingungen zur Validierung von Netzwerkalgorithmen für Microarray-Daten

Dissertation
zur Erlangung des Grades eines Doktors der Naturwissenschaften
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Markus Ruschhaupt
aus Werther (Westf.)

München, den 22. Jannuar 2008

1. Berichterstatter: Prof. Dr. Ulrich Mansmann
2. Berichterstatter: Prof. Dr. Jörg Rahnenführer
Tag des Rigorosums: 06. Juni 2008

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Zusammenfassung

Microarray-Experimente werden in letzter Zeit vermehrt genutzt, um Netzwerke der Gen-Gen-Interaktion zu generieren. Verschiedene Ansätze sind vorgeschlagen worden, diese Netzwerke zu erstellen, wobei diese Methoden auf Grund der vorliegenden Daten heuristische Komponenten enthalten und für das beschriebene Vorgehen selten eine theoretische Begründung geliefert wird. Auch die Validierung eines solchen Algorithmus ist häufig unzureichend, denn geeignete Methoden für die Simulation von Strukturen, die die biologischen Zusammenhänge reflektieren, sind nicht gegeben.

Diese Arbeit beschäftigt sich mit Problemen, die in Validierungsstudien auftreten. Der Startpunkt einer Validierungsstudie ist ein ungerichteter Graph, der biologische Strukturen repräsentieren soll. Die Frage ist nun, welche graphischen Strukturen vorgegeben werden sollen. In dieser Arbeit wird motiviert, dass es sinnvoll ist, Graphen zu benutzen, die aus Microarray-Daten geschätzt worden sind.

Nachdem der Graph gewählt worden ist, werden Daten einer multivariaten Normalverteilung erzeugt, die durch eine zufällige Kovarianzmatrix charakterisiert ist. Die zu der Kovarianzmatrix inverse Matrix wird als Präzessionsmatrix bezeichnet und zur Berechnung der partiellen Korrelationen benutzt. Diese Matrix muss symmetrisch und positiv definit sein, aber zusätzlich soll sie auch die durch den Graphen gegebenen Strukturen repräsentieren. Im Detail wird durch eine nicht vorhandene Kante im Graphen gefordert, dass der zugehörige Eintrag in der Matrix Null ist. Häufig werden diagonaldominante Matrizen für die Erzeugung solcher Matrizen gewählt. Aber wenn man Matrizen betrachtet, die aus Microarray-Daten geschätzt worden sind, so sind diese nicht notwendigerweise diagonaldominant. In dieser Arbeit wird ein neuer Ansatz vorgestellt, der es ermöglicht, symmetrische, positiv definite Matrizen mit Nebenbedingungen zu erzeugen. Dabei wird eine neu definierte Klasse von Graphen benutzt.

Die hier vorgestellte Methode beruht auf der Moralisierung eines Graphen, wobei die Zusammenhänge zwischen einem gerichteten azyklischen Graphen, der Moralisierung dieses Graphen und den beiden die Graphen repräsentierenden Matrizen genutzt werden, um eine Matrix mit den gewünschten Eigenschaften zu erzeugen. Ein gerichteter azyklischer Graph wird moralisiert, indem die gerichteten Kanten durch ungerichtete Kanten ersetzt werden und zusätzlich die Eltern eines jeden Knotens paarweise miteinander verbunden werden. In dieser Arbeit wird die Klasse der Graphen eingeführt, die Resultat einer solchen Moralisierung sein können; Graphen dieser Klasse werden *prämoralisierbar* genannt. Es zeigt sich, dass nicht jeder Graph prämoralisierbar ist. Diese Eigenschaft ist aber notwendig

für den vorgestellten Algorithmus zur Erzeugung der Matrizen mit Nebenbedingungen. Aus diesem Grund wird eine empirische Studie durchgeführt, die zeigt, dass ein Großteil der aus Microarray-Daten geschätzten Graphen auch prämoralisierbar ist.

Die beschriebene Umkehrung des Moralisierungsvorganges ist der zentrale Schritt bei der Erstellung der Matrizen mit Nebenbedingungen. In dieser Arbeit wird ein Algorithmus vorgestellt, der für einen beliebigen ungerichteten Graphen H entscheidet, ob dieser prämoralisierbar ist, und in einem solchen Fall einen gerichteten azyklischen Graphen G findet, dessen Moralisierung H ist. Alle in dieser Arbeit vorkommenden empirischen Untersuchungen basieren auf diesem Algorithmus.

Die erzeugten Matrizen sollen als partielle Korrelationsmatrizen für die Validierung der Netzwerkalgorithmen genutzt werden. Hierzu wird der vorgestellte Algorithmus an ein Optimierunsverfahren gekoppelt, um symmetrische, positiv definite Matrizen mit Nebenbedingungen zu erstellen, deren Diagonalelemente identisch 1 sind und für die die nicht als Null vorgegebenen Werte nahe 1 beziehungsweise -1 liegen; gerade solche Matrizen sind für eine Validierungs von Netzwerkalgorithmen nützlich.

Die Arbeit schließt mit praktischen Anwendungen. Zuerst wird eine Validierung eines bekannten Algorithmus zum Schätzen von Netzwerken durchgeführt. Es zeigt sich, dass die Wahl der graphischen Strukturen und die Art der Matrixgenerierung einen großen Einfluss auf die Resultate einer solchen Untersuchung haben. Im letzten Teil der Arbeit wird ein Ansatz vorgestellt, mit dem man graphische Strukturen, die aus Microarray-Daten geschätzt worden sind, vergleichen kann, um signifikante Unterschiede zu finden. Hier kann der beschriebene Optimierungsalgorithmus auch seine Anwendung finden.

Summary

Microarray studies provide a rich source of data which are recently used by scientists to construct gene interaction networks. Different methods for construction have been proposed which are developed on a heuristic basis and do seldom have a theoretical foundation. The validation of an algorithm is often insufficient, because appropriate models for the simulation of graphs which reflect biologically relevant issues are not available.

This thesis considers some questions concerning the set-up of the simulation studies. Starting point is an undirected graph that is supposed to reflect biologically relevant issues, for example the interaction between genes. The question arises what kind of graphs should be used here. We propose to use graphs whose structure can be derived from microarray studies.

After the graph has been chosen, data is generated from a multivariate normal distribution represented by a random covariance matrix. The inverse of the covariance matrix is called precision matrix. This matrix has to be symmetric and positive definite but is also supposed to reflect the biological restrictions given by the graph. In detail, if there is no edge between two nodes in the graph then the corresponding entry in the matrix should be zero. For this purpose, often diagonally dominant matrices are used, but matrices derived from microarray data do not necessarily have this property. We introduce a new algorithm for positive definite matrices with constraints that uses a new class of graphs.

The method we present makes use of the moralisation of a graph and connections between a directed acyclic graph, the moralisation of this graph and the matrices that represent both graphs. A directed acyclic graph can be moralised by exchanging the directed edges with undirected edges and inserting edges between any two parents of a node. In this thesis we introduce the class of undirected graphs that could be the result of such a moralisation and call them *premoralisable* graphs. We show that not every graph is premoralisable; however, this property is necessary for the proposed algorithm. Therefore we make an empirical study and find out that most of the graphs that are estimated from microarray data are indeed premoralisable.

This reversion of the moralisation is the important step for the construction of the matrices with constraints. We introduce an algorithm that is able to determine for an arbitrary graph H if this graph is premoralisable. In such a case the algorithm constructs a directed acyclic graph G, whose moralisation is H. The empirical study is based on this algorithm.

We want to use the generated matrices as partial correlation matrices for the validation

of network constructing algorithms. Therefore, the proposed algorithm is used together with an optimisation algorithm in order to construct matrices with constraints, such that all diagonal elements are 1 and additionally the non zero elements of the matrix are close to -1 or 1. Such matrices are very useful for validation studies.

In the last part of the thesis we validate an algorithm that constructs a network from microarray data. We find out that indeed the altering of the underlying structures and the way how to generate the correlation matrix can affect the outcome of the study heavily. We also introduce a new approach for the comparison of graph structures that arise from network constructing algorithms. The proposed optimisation strategy for matrices with constraints can be used here as well.

Kapitel 1

Einleitung

In der Molekularbiologie werden Microarray-Experimente seit vielen Jahren benutzt, um das mRNA Profil einer Probe, welche beispielsweise aus Gewebe oder Zellinien gewonnen wird, messen zu können (siehe [69] für eine Zusammenfassung). Jede Probe wird auf einen Microarray-Chip aufgetragen, um so das Transkriptionsverhalten von ca. 30000 Genen messen zu können. Ein komplettes Microarray-Experiment beinhaltet in der Regel zwischen 10 und 500 Chips, die sich häufig in zwei oder mehr Gruppen bezüglich eines vorher definierten Phänotyps einteilen lassen. Das Ziel eines Microarray-Experimentes besteht somit oft darin, Gene zu finden, die zwischen den vorher definierten Gruppen unterschiedlich exprimiert sind.

In jüngerer Zeit werden die Daten einer Microarray-Studie vermehrt benutzt, um strukturelle Zusammenhänge zwischen den Genen zu erfassen und so Netzwerke aus Genen zu generieren. Hierzu sind verschiedene Ansätze vorgestellt worden, die direkte beziehungsweise indirekte Interaktionen modellieren. Der Nutzen eines Netzwerkes sei am Beispiel von Herzerkrankungen erläutert: Bei Herzerkrankungen unterscheidet man unter anderem zwischen dilatativer und ischämischer Kardiomyopathie. Es wird angenommen, dass diese Unterscheidung für die weitere Behandlung der Patienten von großer Wichtigkeit ist [1], wobei die herkömmlichen Verfahren zur Diagnose mit Fehlern behaftet sind [73]. Deshalb wird die Hoffnung in die Bestimmung und Unterscheidung dieser Kardiomyopathie-Subtypen mit Hilfe von Microarray-Daten gelegt. Während in einigen wenigen Publikationen von großen Unterschieden berichtet wird wie beispielsweise bei Kittleson[39], können diese in anderen Veröffentlichungen nicht bestätigt werden [67]. Es kann, zum Beispiel nach einer Gesamtbetrachtung der Studien, aber eher davon ausgegangen werden, dass man durch univariate Verfahren und Klassifikationsansätze keine wirkliche Trennung zwischen den beiden Gruppen vornehmen kann. Durch die Erzeugung eines Interaktionsnetzwerkes ist es jedoch möglich, signifikante Unterschiede bei dem Interaktionsverhalten der Gene zwischen diesen Gruppen zu finden (siehe Kapitel 5).

Im obigem Beispiel bestehen die Netzwerke aus ungerichteten Graphen. Sie werden benutzt, um Unterschiede in der Interaktionsstruktur zwischen zwei Patientenkollektiven zu untersuchen. Es liegt nicht im Fokus der Untersuchung, den Interaktionsablauf zu verstehen. Dies bedeutet, man weiß nicht, in welcher Reihenfolge die Gene miteinander in-

teragieren. Dafür ist ein ungerichteter Graph nicht geeignet. Ist man an dem Ablauf der Geninteraktion interessiert, so sollte ein gerichteter Graph erzeugt werden, wobei hier allerdings ein entsprechendes Experiment vorliegen sollte. Als Beispiel für die Erstellung eines direkten Expressionsnetzwerkes sei hier die Arbeit von Markowetz[49] genannt, die *Nested Effect Modelle* definiert und benutzt. Für diesen Ansatz liegen Daten zu Grunde, die auf siRNA-Experimenten beruhen. Vereinfacht gesprochen wird bei einem siRNA-Experiment, auch als ein *knock-down* Experiment bezeichnet, die Transkription eines Gens reduziert - im besten Fall bis zu dem Punkt, an dem keine Transkription des Gens mehr vorhanden ist. Es werden dann die Expressionsunterschiede untersucht, die sich durch die Reduktion des Gens ergeben, das bedeutet man misst beispielsweise mit Hilfe eines Microarray-Chips die Expression vieler Gene vor der Behandlung mit siRNA und danach. Das gesamte Experiment wird wiederholt, wobei verschiedene Gene in ihrer Expression reduziert werden. Aus den gewonnenen Daten wird mit Hilfe der *Nested Effect Modelle* ein Netzwerk erzeugt. Dieses beruht darauf, dass für zwei Gene A und B die Menge der durch die Reduktion von A veränderten Gene M_A ganz enthalten ist in der Menge der durch die Reduktion von B veränderten Gene M_B, falls B auf A eine Interaktion ausübt. Es ergibt sich ein gerichteter Graph.

Bei den meisten Microarray-Datensätzen liegen keine Messungen vor, bei denen die Expressionen von Genen gestört worden sind. In einem solchen Fall ist es sinnvoll, einen ungerichteten Graphen zu erzeugen, beispielsweise basierend auf Korrelationsbeobachtungen. In einem *relevance network* sind zwei Knoten, die in diesem Fall Gene oder Spots auf dem Microarray repräsentieren, miteinander verbunden, falls die entsprechenden Variablen eine von Null abweichende Korrelation haben [10]. Solche Graphen sind leicht zu erstellen, da für jede einzelne Korrelationsberechnung nur zwei Variablen benutzt werden, so dass immer mehr Beobachtungen als Variablen gegeben sind. Der Nachteil solcher Korrelationsnetzwerke besteht darin, dass sie nicht nur direkte, sondern auch indirekte Interaktionen abbilden, an denen man weniger interessiert ist [64]. Zwei Gene können auch eine hohe Korrelation haben, falls sie nur indirekt interagieren, also beispielsweise beide durch einen Transkriptionsfaktor reguliert werden.

Eine bessere Vorstellung der Vorgänge erhält man, wenn man die Korrelation zweier Variablen gegeben alle übrigen Variablen betrachtet. Ansätze für die Erstellung von Netzwerken, die auf diesen Überlegungen beruhen, sind beispielsweise Schäfer und Strimmer[63], [64], Meinshausen[50] und Dobra[18]. Bei diesen Ansätzen wird davon ausgegangen, dass die Expression der Gene normalverteilt ist, zu einem Mittelwert μ und zu einer Kovarianzmatrix Σ. Man möchte nun $\Omega = \Sigma^{-1}$, die *Präzessionsmatrix*, schätzen, da diese die partielle Korrelation beschreibt, das bedeutet die Korrelation zwischen zwei Variablen gegeben alle anderen Variablen. Ähnlich wie bei einem *relevance network* wird ein Graph dann erzeugt, indem man alle Variablenpaare, bei denen sich die partielle Korrelation signifikant von Null unterscheidet, durch eine Kante verbindet.

Die empirische Schätzung der Präzessionsmatrix ist nicht möglich, falls es mehr Variablen als Beobachtungen gibt. Dies ist aber bei einem Microarray-Experiment immer der Fall, da die Variablen den Genen entsprechen und die Beobachtungen den Proben, also der Anzahl der zur Verfügung stehenden Chips. Somit werden zur Schätzung der Matrix

Σ beziehungsweise Ω Verfahren benutzt, die heuristische Elemente enthalten. Für solche Ansätze sind Validierungsverfahren durch Simulationen von großer Bedeutung, denn eine theoretische Begründung für das beschriebene Vorgehen ist selten gegeben. Mit Hilfe von Validierungsverfahren soll vor allem untersucht werden, ob die graphischen Strukturen, die man vorgegeben hat, mit den Stukturen übereinstimmen, die vom Algorithmus aus simulierten Daten erzeugt worden sind. Es gibt verschiedene Möglichkeiten, diese Strukturen zu vergleichen, von denen einige in dieser Arbeit erwähnt werden. Weitere Ziele eines Validierungsverfahrens bestehen darin, zu testen, wie robust die Resultate sind. Das heißt, es wird untersucht, wie weit sich die Resultate eines Algorithmus unterscheiden, wenn man gewisse Untermengen der gesamten Daten betrachtet. Hat man beispielsweise 500 simulierte Datenpunkte, so kann man untersuchen, wie sich die Struktur ändert, wenn man *Bootstrap* Ziehungen des gesamten Datensatzes betrachtet. Mit Hilfe von Simulationsansätzen ist es auch möglich, Fallzahlberechnungen und Powerschätzungen durchzuführen, denn für einen gegebenen Algorithmus lassen sich Sensitivität und Spezifität für eine gegebene Anzahl von Proben abschätzen.

Allgemein sollen in einem Simulationsansatz multivariat normalverteilte Daten mit bekannten Abhängigkeitsstrukturen erzeugt werden. Diese Abhängigkeitsstrukturen sind gegeben durch einen ungerichteten Graphen $H = (V, E)$ mit $n = |V|$. Das bedeutet, die Expressionsdaten sind normalverteilt zu den Parametern μ und Σ, und zwischen H und der zu Σ inversen Matrix Ω besteht der Zusammenhang, dass die Einträge der Präzessionsmatrix Null sind, zu denen es keine Kante im Graphen H gibt. Die so erzeugten Daten können genutzt werden, um einen Algorithmus zu validieren, der ein Netzwerk aus diesen Daten schätzt, denn die dem Modell zu Grunde liegende Struktur ist bekannt und so mit der geschätzten Struktur vergleichbar. Eine Schwierigkeit dieses Simulationsansatzes liegt in der Bereitstellung der Matrix Σ beziehungsweise der inversen Matrix $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$. Ω muss nach obigem Ansatz als Präzessionsmatrix zu einem vorgegebenen Graphen $H = (V, E)$ die folgenden zwei Bedingungen erfüllen:

B1: Ω ist positiv definit

B2: $\{v_i, v_j\} \notin E \Rightarrow \omega_{ij} = 0$

Die zweite Bedingung gibt hierbei an, dass die Elemente der Matrix, für die keine Kante in dem vorgegebenen Graphen vorhanden ist, auf Null gesetzt werden müssen.

Es gibt verschiedene Ansätze, die sich mit dem Erzeugen solcher Matrizen Ω befassen. Meinshausen[50] und Schäfer[64],[63] benutzten für die Validierungen ihrer Algorithmen diagonaldominante Matrizen. Dies ist der einfachste Ansatz, um Matrizen mit den obigen zwei Eigenschaften zu erzeugen. In Simulationsansätzen sollten aber nicht nur die Punkte B1 und B2 erfüllt werden, sondern es ist von Vorteil, wenn man zusätzlich in gewissem Rahmen auch die Stärke der Korellation vorgeben kann. Da bei den meisten Validierungsansätzen nur die Anzahl der falsch negativen, falsch positiven und richtig positiven Resultate gezählt wird, ist es für eine möglichst einfache Auswertung der Daten wünschenswert, dass man alle vorhandenen Kanten mit einer gleichen möglichst starken Korrelation belegt. Damit haben die Kanten die gleiche a priori Wahrscheinlichkeit, vom Algorithmus gefunden

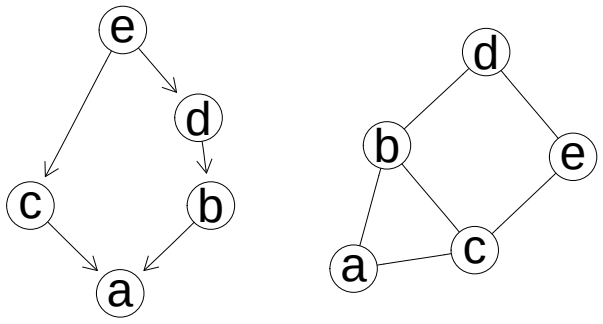

Abbildung 1.1: Beispiel eines gerichteten Graphen und der zugehörigen Moralisierung.

zu werden. Das ist mit diagonaldominanten Matrizen nicht möglich. Diagonaldominante Matrizen bilden zudem nur einen Teil aller positiv definiten Matrizen [57], und auch bei einem Teil der aus Microarray-Daten geschätzten Matrizen liegt keine Diagonaldominanz vor (siehe Kapitel 5). Ein neuer Ansatz zum Erstellen einer oben beschriebenen Matrix Ω ist somit wünschenswert.

In dieser Arbeit wird eine neuer Algorithmus vorgestellt, mit dem es möglich ist, positiv definite Matrizen mit Nebenbedingungen zu simulieren. Der Kernpunkt des neuen Simulationsansatzes sind moralisierte Graphen. Ein gerichteter azyklischer Graph wird moralisiert, indem man die gerichteten Kanten durch ungerichtete Kanten ersetzt und zusätzlich die Eltern eines jeden Knotens paarweise durch eine ungerichtete Kante verbindet (siehe Abbildung 1.1).

Die Moralisierung eines Graphen wurde von Dawid und Lauritzen [16] als ein Hilfsmittel eingeführt, um Markov-Eigenschaften eines gerichteten Graphen mit Hilfe der zugehörigen Moralisierung zu definieren. Für die Erstellung von positiv definiten Matrizen mit Nebenbedingungen wurden Moralisierungen bisher nicht benutzt. Aus den Ergebnissen von Dawid und Lauritzen lassen sich aber Zusammenhänge ableiten, die für die Erzeugung von positiv definiten Matrizen mit Nebenbedingungen nützlich sind. Dafür sei $G = (V, \overrightarrow{E})$ ein gerichteter azyklischer Graph und sei K_G eine Matrix, die G repräsentiert, das bedeutet, die Einträge der Matrix K_G sind Null, falls die zum Eintrag korrespondierende Kante in dem Graphen nicht vorhanden ist. Man bildet dann

$$\Omega := (D - K_G)^t \cdot (D - K_G)^t,$$

wobei hier D eine beliebige Diagonalmatrix mit $d_{ii} \neq 0$ für alle i ist. Es gilt, dass Ω positiv definit ist und die Moralisierung $H = (V, E)$ von G repräsentiert. Das bedeutet, falls eine Kante in H nicht vorhanden ist, so ist der entsprechende Eintrag in der Matrix gleich Null (siehe Abbildung 1.2).

Mit Hilfe dieses Zusammenhanges wird in Kapitel 4 der Arbeit ein Simulationsansatz für positiv definite Matrizen mit Nebenbedingungen eingeführt, der aus folgenden Schritten besteht. Gegeben ist hierbei ein ungerichteter Graph H.

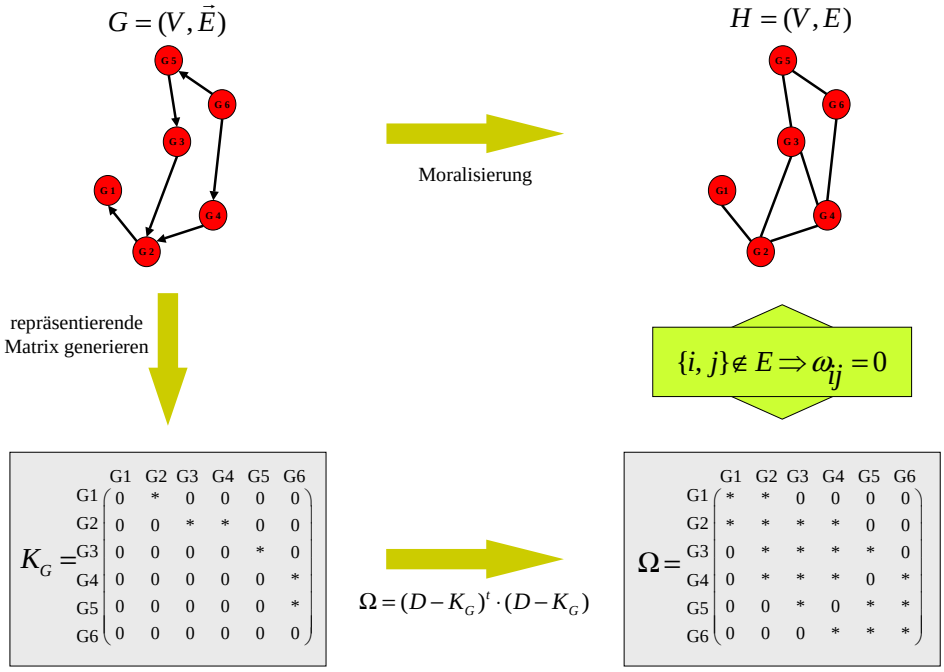

Abbildung 1.2: Die Abbildung zeigt die Zusammenhänge zwischen der Moralisierung eines Graphen und den Null-Elementen einer positiv definiten Matrix.

1. Finde zu H einen gerichteten azyklischen Graphen G, so dass die Moralisierung von G dem vorgegebenen Graphen H entspricht.
2. Erstelle eine Matrix K_G, die den Graphen G repräsentiert.
3. Besetze die Nicht-Null Elemente in K_G.
4. Erstelle eine Diagonalmatrix D mit vollem Rang.
5. Bilde $\Omega = (D - K_G)^t \cdot (D - K_G)$.

Die so erstellte Matrix Ω erfüllt dann die Bedingungen B1 und B2.

Der schwierige Schritt in diesem Algorithmus ist der erste, die Umkehrung des Moralisierungsvorganges. Es soll für einen ungerichteten Graphen H ein gerichteter azyklischer Graph G gefunden werde, dessen Moralisierung dem vorgegebenen Graphen H entspricht. Ein solcher Graph wird in dieser Arbeit *Prämoralisierung* von H genannt.
Kapitel 3 dieser Arbeit befasst sich allgemein mit prämoralisierbaren Graphen. Eine Prämoralisierung wird definiert und es wird gezeigt, dass nicht jeder Graph prämoralisierbar ist. Zusätzlich werden einige notwendige und einige hinreichende Kriterien formuliert, mit deren Hilfe man testen kann, ob ein Graph prämoralisierbar ist. Den Kernpunkt des Kapitels bildet ein Algorithmus, der es erlaubt, für einen ungerichteten Graphen H eine

Prämoralisierung zu finden. Es wird bewiesen, dass der Algorithmus genau dann einen prämoralisierbaren Graphen findet, falls ein solcher Graph existiert. Es ergibt sich somit eine notwendige *und* hinreichende Bedingung für die Eigenschaft, dass ein Graph eine Prämoralisierung besitzt.

Mit Hilfe einer Prämoralisierung ist man dann durch den obigen Ansatz in der Lage, die gewünschte Matrix Ω zu erzeugen. Freiraum gibt es noch bei Schritt 3 und 4, der Besetzung der Matrizen K_G und D. Die Matrix K_G hat vorgeschriebene Nullstellen, und die übrigen Einträge können beliebig besetzt werden. Für den in Kapitel 4 vorgestellten Validierungsansatz von Netzwerkalgorithmen ist folgende Strategie sinnvoll: Die freien Einträge werden mit Hilfe eines Optimierungsverfahrens so besetzt, dass die erzeugte Matrix Ω bezüglich der Frobeniusnorm einen möglichst kleinen Abstand zu einer vorgegebenen, nicht notwendigerweise positiv definiten Matrix A hat, das heißt die Einträge der Matrix K_G werden so gewählt, dass $\|K_G^t \cdot K_G - A\|$ minimiert wird. Hierzu werden zwei verschiedene Optimierungsverfahren angewandt und die Resultate verglichen. Falls A entsprechend besetzt wird, ist es somit insbesondere möglich, die Bedingung zu erfüllen, dass alle Einträge der Matrix eine vom Betrag hohe partielle Korrelation besitzen. Aber auch andere Besetzungen von A sind möglich, die unter Umständen andere Validierungsansätze nach sich ziehen können.

Wie in Kapitel 3 gezeigt, ist nicht jeder Graph prämoralisierbar. Da diese Eigenschaft aber für den in Kapitel 4 beschriebenen Simulationsansatz notwendig ist, muss untersucht werden, wie groß der Anteil der prämoralisierbaren Graphen an allen ungerichteten Graphen ist. In dieser Arbeit beschränkt sich die Untersuchung auf die Menge der graphischen Strukturen, die durch Algorithmen aus Microarray-Daten geschätzt worden sind, denn solche Strukturen eignen sich als Ausgangspunkt für Simulationsansätze. In Kapitel 5 wird gezeigt, dass der Anteil der prämoralisierbaren Graphen an den Graphen, die aus Microarray-Daten geschätzt worden sind, sehr groß ist. Zudem wird gezeigt, dass die Anzahl der Knoten eines Graphen mit der Prämoralisierungseigenschaft korreliert.

Mit Hilfe der prämoralisierbaren Graphen, die aus acht Microarray-Datensätzen geschätzt worden sind, wird in Kapitel 5 exemplarisch ein Algorithmus von Schäfer und Strimmer[64] mit dem vorgestellten Simulationsansatz validiert. Es ergeben sich hierbei starke Unterschiede zu den Simulationsergebnissen aus [64].

Mit dem vorgestellten Simulationsansatz ist es aber nicht nur möglich, Netzwerkalgorithmen zu validieren. Als Beispiel und als Vorbereitung für die übrigen Abschnitte wird im ersten Abschnitt von Kapitel 5 untersucht, ob es besser ist, alle Spots eines Microarray-Chips für die Erzeugung eines Netzwerkes zu nutzen, oder ob man die Spots vorher zu Genen zusammenfassen sollte. Auf einem Microarray-Chip sind normalerweise viele Gene durch mehrere Spots repräsentiert. Diese Spots sollten die gleiche Genexpression messen, da sich Unterschiede vor allem durch eine technische Varianz ergeben sollten, die normalerweise geringer ist als eine biologische Varianz. Wenn man ein Netzwerk erstellt und alle Spots verwendet, so gibt es im finalen Netz verschiedene Knoten, die das gleiche Gen repräsentieren. Die Interpretation eines solchen Netzes kann sich als sehr schwierig erweisen, falls man gegensätzliche Resultate bekommt. Ein mögliches Resultat einer solchen Netzwerkkonstruktion zeigt Schaubild 1.3. Hier wird ein Netzwerk auf den Spots eines Microarray-Chips

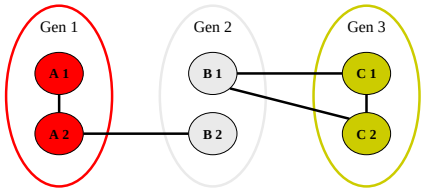

Abbildung 1.3: Schaubild eines Netzwerkes, bei dem jeder Knoten einen Spot auf einem Microarray repräsentiert und unterschiedliche Knoten das gleiche Gen repräsentieren. Knoten, die das gleiche Gen repräsentieren, haben die gleiche Farbe. Kanten stehen für gefundene partielle Korrelationen, die signifikant von Null abweichen.

erstellt, wobei Spots gleicher Farbe das gleiche Gen repräsentieren. Ein Spot von Gen 2 interagiert mit beiden Spots von Gen 3, während der zweite Spot von Gen 2 mit einem Spot von Gen 1 interagiert. Hieraus lässt sich nicht eindeutig ableiten, ob nun Gen 2 mit Gen 1 oder Gen 3 oder mit beiden Genen interagiert. Die Interpretation der Daten wird einfacher, falls man nur einen Knoten pro Gen in dem finalen Netzwerk hat. Reduziert man aber die Daten auf nur einen Spot pro Gen, ist die Frage, ob diese noch ausreichen, die Struktur wiederzufinden. Diese Frage wird in Kapitel 5 dieser Arbeit mit einem vorgestellten Simulationsansatz untersucht. Interessanterweise ist eine Schätzung des Netzes auf Genebene, wenn nur ein Spot pro Gen benutzt wird, nicht nur gleichwertig, sondern besser als eine Schätzung auf Spotebene.

In der diese Arbeit abschließenden Untersuchung in Kapitel 5 werden deshalb auch Netzwerke verglichen, die auf Genebene und nicht auf Spotebene geschätzt worden sind. Wie zu Anfang erwähnt, ist der Vergleich von Strukturen auf Genexpressionsdaten ein mögliches Resultat einer Netzwerkuntersuchung. Leider ist dem Vergleich von graphischen Strukturen bei Microarray-Daten nur geringe Aufmersamkeit gewidmet worden, beispielsweise in einer Arbeit von Balasubramanian[2]. In der vorliegenden Arbeit wird ein weiteres Verfahren vorgestellt, welches auf einem parametrischen Bootstrap beruht. Mit Hilfe des parametrischen Bootstraps werden schließlich unter anderem für einen Microarray-Datensatz, der den eingangs erwähnten Vergleich zwischen dilatativer und ischämischer Kardiomyopathie untersucht [33], die Pathways extrahiert, die sich in ihrer Struktur signifikant zwischen diesen beiden Patientenkollektiven unterscheiden. Die gewonnenen Resultate lassen sich teilweise durch vorhandene Erkenntnisse über Kardiomyopathie erklären und liefern einen Ansatzpunkt für weiterführende Studien. Da für den parametrischen Bootstrap wieder Daten aus einer multivariaten Normalverteilung mit Nebenbedingungen erzeugt werden müssen, besteht hier auch die Möglichkeit, den in dieser Arbeit vorgestellten Algorithmus einzusetzen.

Kapitel 2

Methodische Grundlagen

In Kapitel 2 werden mathematische und statistische Grundlagen vorgestellt, die in dieser Arbeit benutzt werden. Ein Schwerpunkt liegt im Bereich der Graphentheorie, wobei hier die Definitionen und Sätze überwiegend aus dem Buch von Diestel [17] übernommen worden sind. Nicht alle in dieser Arbeit vorkommenden Definitionen sind in Kapitel 2 aufgeführt, einige speziellere Definitionen werden in späteren Kapiteln gegeben.

2.1 Grundbegriffe der Graphentheorie

Definition 2.1.1 *Ein Graph ist ein Paar $H = (V_H, E_H)$, wobei V_H eine Menge ist, deren Elemente Knoten von H heißen, und $E_H \subseteq \binom{V_H}{2} := \{\{v, w\} \,|\, v, w \in V_H, v \neq w\}$ eine Menge von 2-elementigen Teilmengen von V_H ist, die als Kanten von H bezeichnet werden.*

Die Knotenmenge eines Graphen, und damit auch die Kantenmenge, kann sowohl endlich als auch unendlich sein. Entsprechend heißt dann auch der Graph endlich oder unendlich. In dieser Arbeit werden nur endliche Graphen behandelt, so dass der Zusatz im weiteren Verlauf weggelassen wird. Manchmal wird ein Graph als ungerichteter Graph bezeichnet, um zu betonen, dass es sich nicht um einen gerichteten Graphen (Definition siehe unten) handelt. Folgende Definitionen sind für einen Graphen elementar und werden vermehrt in Kapitel 3 benutzt.

Definition 2.1.2 *In einem Graphen $H = (V_H, E_H)$ heißen zwei Knoten $v, w \in V_H$ benachbart oder adjazent, falls $\{v, w\} \in E_H$ ist. Man bezeichne mit $N_H(v) \subset V_H$ die Menge der zu einem Knoten v in einem Graphen H benachbarten Knoten. Zwei Knoten $v, w \in V_H$ haben einen gemeinsamen Nachbarn, falls $N_H(v) \cap N_H(w) \neq \emptyset$.*

Ein Knoten $v \in V_H$ heißt inzident mit einer Kante $e \in E_H$, falls $v \in e$ gilt. Für einen Knoten v bezeichnet die Menge $I_H(v) \subseteq E_H$ die Menge der mit v inzidenten Kanten:

$$I_H(v) := \{e \in E_H \,|\, v \in e\}.$$

Als Grad eines Knotens $v \in V$ in $H = (V_H, E_H)$ bezeichnet man die Anzahl der mit v inzidenten Kanten

$$\deg_H(v) := |\, I_H(v) \,|\,.$$

Mit $IN_H(v) \subset E_H$ wird die Menge der Kanten bezeichnet, die sich zwischen zu v adjazenten Knoten befinden.

$$IN_H(v) := \{\{x, y\} \in E_H \,|\, x, y \in N_H(v)\}$$

Neben den ungerichteten Graphen werden in dieser Arbeit auch gerichtete Graphen behandelt. Gibt es bei ungerichteten Graphen nur Nachbarschaftsverhältnisse, so werden bei gerichteten Graphen zusätzlich *Kinder* und *Eltern* eines Knotens definiert.

Definition 2.1.3 *Ein gerichteter Graph ist ein Paar $G = (V_G, \overrightarrow{E_G})$ mit V_G als Knotenmenge und*

$$\overrightarrow{E_G} \subseteq \{(v, w) \,|\, v, w \in V_G, v \neq w\}$$

als Menge gerichteter Kanten, so dass gilt: Ist $(v, w) \in \overrightarrow{E_G}$ dann ist $(w, v) \notin \overrightarrow{E_G}$. Ist $H = (V_H, E_H)$ ein ungerichteter Graph mit $V_H = V_G$ und

$$E_H = \{\{v, w\} \in \binom{V_G}{2} \,|\, (v, w) \in \overrightarrow{E_G} \vee (w, v) \in \overrightarrow{E_G}\}$$

dann heißt H der G zugrundeliegende ungerichtete Graph.

In dieser Arbeit werden also nur gerichtete Graphen betrachtet, die keine Doppel- oder Mehrfachkanten besitzen. Somit treten nur gerichtete Graphen G auf, bei denen keine zwei Knoten $v, w \in V_G$ existieren mit $(v, w) \in E_G$ und $(w, v) \in E_G$. Zudem gibt es auch keine Kanten, wo Start- und Zielknoten identisch sind, das bedeutet es existiert kein $v \in V_G$ mit $(v, v) \in E_G$.

Definition 2.1.4 *Sei $G = (V_G, \overrightarrow{E_G})$ ein gerichteter Graph. Zwei Knoten in G heißen adjazent beziehungsweise ein Knoten und eine Kante inzident, wenn dies in dem G zugrundeliegenden ungerichteten Graphen der Fall ist. Analog werden N_G,I_G,IN_G und $\deg_G$ für einen gerichteten Graphen über den zugrundeliegenden ungerichteten Graphen definiert.*

Ein Knoten $w \in V_G$ heißt Elternteil eines Knotens $v \in V_G$, falls $(w, v) \in \overrightarrow{E_G}$. Mit $P_G(v) = \{w \in V_G \,|\, (w, v) \in \overrightarrow{E_G}\}$ werden alle Eltern eines Knoten v in G bezeichnet. Ein Knoten $w \in V_G$ heißt Kind eines Knotens $v \in V_G$, falls $v \in P_G(w)$. Mit $C_G(v) = \{w \in V_G \,|\, (v, w) \in \overrightarrow{E_G}\}$ werden alle Kinder eines Knoten v in G bezeichnet. Für einen Knoten $v \in V_G$ definiert man zusätzlich die Menge der eingehenden und ausgehenden Kanten:

$$v^+ := \{(v, w) \in \overrightarrow{E_G} \,|\, w \in V_G\} \text{ und } v^- := \{(w, v) \in \overrightarrow{E_G} \,|\, w \in V_G\}$$

Häufig werden bei einem gerichteten oder ungerichteten Graphen Knoten oder Kanten entfernt. Der resultierende Graph ist dann ein *Teilgraph* des ursprünglichen Graphen. Da solche Operationen im Folgenden vermehrt benutzt werden, wird hierfür eine eigene Notation eingeführt.

Definition 2.1.5 *Ein Graph $H_1 = (V_{H_1}, E_{H_1})$ heißt Teilgraph eines zweiten Graphen $H_2 = (V_{H_2}, E_{H_2})$, wenn $V_{H_1} \subseteq V_{H_2}$ und $E_{H_1} \subseteq E_{H_2}$ gilt. Analog heißt ein gerichteter Graph $G_1 = (V_{G-1}, \overrightarrow{E_{G_1}})$ Teilgraph eines gerichteten Graphen $G_2 = (V_{G_2}, \overrightarrow{E_{G_2}})$, falls $V_{G_1} \subseteq V_{G_2}$ und $\overrightarrow{E_{G_1}} \subseteq \overrightarrow{E_{G_2}}$.*

Definition 2.1.6 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph. Für $V' \subseteq V_H$ ist der Teilgraph $H - V'$ definiert durch $H - V' := (V_H \backslash V', \{e \in E_H \,|\, e \subseteq V_H \backslash V'\})$. Für $E' \subseteq E_H$ ist der Teilgraph $H - E'$ definiert durch $H - E' := (V_H, E_H \backslash E')$. Man spricht dann auch vom Entfernen der Knotenmenge V' beziehungsweise der Kantenmenge E' aus dem Graphen H. Ist $E' \subseteq \binom{V_H}{2}$ so definiert man $H \cup E' := (V_H, E_H \cup E')$.*

Ist $G = (V_G, \overrightarrow{E_G})$ ein gerichteter Graph und $V' \subseteq V_G$, so ist der Graph $G - V'$ definiert durch $G - V' := (V_G \backslash V', \{(v, w) \in \overrightarrow{E_G} \,|\, v, w \in V_G \backslash V'\})$. Für $E' \subseteq \overrightarrow{E_G}$ sei außerdem $G - E' := (V_G, \overrightarrow{E_G} \backslash E')$ definiert. Ist $E' \subseteq \{(v, w) \,|\, v, w \in V_G \wedge v \neq w \wedge (w, v) \notin \overrightarrow{E_G}\}$ so definiert man $G \cup E' := (V_G, \overrightarrow{E_G} \cup E')$.

Sowohl bei ungerichteten als auch bei gerichteten Graphen gibt es Pfade und Zykel. Pfade beschreiben einen *Weg* von einem Startknoten zu einem Zielknoten, der nur über im Graphen vorhandene Kanten verläuft, wobei in einem gerichteten Graphen die Ausrichtung der Kanten berücksichtigt werden muss. Ein Zykel ist ein Pfad, bei dem zusätzlich noch eine Kante zwischen Ziel- und Startknoten vorhanden ist. Eine Zusammenhangskomponente eines Graphen ist ein Teilgraph, bei dem alle vorhandenen Knoten paarweise durch einen Pfad verbunden werden können.

Definition 2.1.7 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph und $n \geq 2$. Ein Pfad in H ist ein Teilgraph P von H mit einer Menge $v_0, \ldots, v_{n-1} \in V_H$ von n paarweise disjunkten Knoten und der Kantenmenge $\{\{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{n-2}, v_{n-1}\}\}$. Man schreibt $P = v_0, \ldots, v_{n-1}$ und sagt, P ist ein Pfad von v_0 nach v_{n-1}. Seien $A, B \subseteq V_H$. Ein Pfad $P = v_0, \ldots, v_{n-1}$ verläuft von A nach B, falls $v_0 \in A$ und $v_{n-1} \in B$. Sei $C \subseteq V_H$. Ein Pfad $P = v_0, \ldots, v_{n-1}$ verläuft durch C, falls ein $i \in \{1, \ldots, n-2\}$ existiert mit $v_i \in C$.*

Sei $G = (V_G, \overrightarrow{E_G})$ ein gerichteter Graph und $n \geq 2$. Ein Pfad in G ist ein Teilgraph P von G mit einer Menge $v_0, \ldots, v_{n-1} \in V_G$ von n paarweise disjunkten Knoten und der Kantenmenge $\{(v_0, v_1), (v_1, v_2), \ldots, (v_{n-2}, v_{n-1})\}$. Man schreibt $P = v_0, \ldots, v_{n-1}$ und sagt, P ist ein Pfad von v_0 nach v_{n-1}. Wie bei einem ungerichteten Graphen definiert man, dass ein Pfad von A nach B oder durch C verläuft.

Definition 2.1.8 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph. Sei $V' \subseteq V_H$, so dass für alle $v, w \in V'$ ein Pfad von v nach w in H existiert und für alle $v \in V'$ und $w \in V_H \backslash V'$ kein Pfad von v nach w in H existiert. Dann wird der Teilgraph $H - (V_H \backslash V')$ als Zusammenhangskomponente von H bezeichnet.*

Es ist einfach zu zeigen, dass sich ein Graph immer in paarweise disjunkte Zusammenhangskomponenten zerlegen lässt.

Definition 2.1.9 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph und sei $n \geq 3$. Der Teilgraph von H bestehend aus einem Pfad $P = v_0, \ldots, v_{n-1}$ zusammen mit der Kante $\{v_{n-1}, v_0\}$ wird als Zykel bezeichnet und man schreibt $Z = v_0, \ldots, v_{n-1}, v_0$. Für alle $i, j \in \{0, \ldots, n-1\}$ heißen zwei Knoten v_i und v_j im Zykel benachbart, falls $j = (i+1) \bmod n$ oder $i = (j+1) \bmod n$. Ein Zykel ohne Abkürzung in $H = (V_H, E_H)$ ist ein Zykel $Z = v_0, \ldots, v_{n-1}, v_0$ in H mit $n \geq 4$, so dass für alle Knoten v_i, v_j des Zykels gilt, dass $\{v_i, v_j\} \notin E_H$, falls v_i und v_j nicht im Zykel benachbart sind.*

Sei $G = (V_G, E_G)$ ein gerichteter Graph und sei $n \geq 3$. Der Teilgraph von G bestehend aus einem Pfad $P = v_0, \ldots, v_{n-1}$ zusammen mit der Kante (v_{n-1}, v_0) wird als Zykel bezeichnet und man schreibt $Z = v_0, \ldots, v_{n-1}, v_0$. Ein gerichteter Graph G heißt azyklisch, wenn er keinen Zykel enthält. Ein solcher Graph wird auch als directed acyclic graph (DAG) bezeichnet.

Weil ein DAG keine Zykel enthält, kann man immer eine *Nummerierung* der Knoten finden, so dass für jeden Knoten v die Kinder von v eine kleinere Nummer haben als v selbst. Aus diesem Grund lässt sich ein DAG durch eine obere Dreiecksmatrix repräsentieren (siehe Kapitel 2.3 über Gaußsche graphische Modelle).

Definition 2.1.10 *Gegeben sei ein gerichteter oder ungerichteter Graph $G = (V_G, E_G)$ mit $|V_G| = n$. Eine Nummerierung von V_G ist eine bijektive Abbildung*

$$\phi : V_G \to \{1, \ldots, n\}.$$

Für $i \in \{1, \ldots, n\}$ heißt i das Label des Knotens $v_i = \phi^{-1}(i)$. Zur Vereinfachung werden häufig bei einer gegebenen Nummerierung die Knoten wie das entsprechende Label bezeichnet.

Enthält ein Graph alle möglichen Kanten, so wird er *vollständig* genannt. Vollständige Graphen spielen im späteren Verlauf eine wichtige Rolle, da durch sie keine Restriktionen bei der Erzeugung von positiv definiten Matrizen vorgegeben werden (siehe Kapitel 4).

Definition 2.1.11 *Ein Graph $G = (V_G, E_G)$ heißt vollständig , wenn $E_G = \binom{V_G}{2}$. Eine Knotenmenge $U \subseteq V_G$ heißt vollständig, wenn für alle $v, w \in U$ mit $v \neq w$ gilt: $\{v, w\} \in E_G$.*

Eine Knotenmenge $U \subseteq V_G$ bildet eine Clique in G, falls sie vollständig ist und maximal, das heißt wenn für alle $v \in V_G \backslash U$ gilt, dass $U \cup \{v\}$ nicht vollständig ist.

2.1.1 Moralisierung eines Graphen

Der Begriff *Moralisierung eines Graphen* wurde von Dawid und Lauritzen[16] eingeführt. Dort ist die Moralisierung eines Graphen ein Hilfsmittel, um Markoveigenschaften eines gerichteten Graphen zu formulieren. Als Beispiel erfüllt ein gerichteter Graph G die *global directed Markov property*, falls alle Teilmengen, die auf der zugehörigen Moralisierung *separieren*, unabhängig sind. In dieser Arbeit wird die Moralisierung ein wichtiges Hilfmittel

bei der Erzeugung von positiv definiten Matrizen mit Nebenbedingungen sein. Hierzu wird die Umkehrung der Moralisierung benötigt (siehe Kapitel 3 und 4). Diese Umkehrung ist einfach zu vollziehen für die Klasse der zerlegbaren Graphen.

Definition 2.1.12 *Sei $H = (V_H, E_H)$ ein Graph. Seien $A, B, C \subseteq V_H$. Die Menge C separiert die Mengen A und B, falls jeder Pfad von A nach B durch C läuft. Man schreibt in diesem Fall $A \bowtie B \,|\, C\,[H]$.*

Definition 2.1.13 *Sei $H = (V_H, E_H)$ ein Graph. Ein Tripel (A, B, C) von disjunkten Teilmengen mit $A, B, C \subseteq V_H$ bildet eine Zerlegung von H, falls gilt:*

1. *$V_H = A \cup B \cup C$*
2. *C separiert A und B*
3. *C ist vollständig in H*

Falls weder A noch B leer sind, heißt die Zerlegung echt.

Definition 2.1.14 *Ein Graph $H = (V_H, E_H)$ heißt zerlegbar, falls der Graph entweder vollständig ist, oder eine echte Zerlegung (A, B, C) besitzt, so dass $H - B$ und $H - A$ zerlegbar sind.*

Eine Charakterisierung der zerlegbaren Graphen erhält man über die perfekten Nummerierungen. Diese Charakterisierung ist wichtig für die späteren Untersuchungen der Zusammenhänge zwischen Zerlegbarkeit und Moralisierung (siehe Kapitel 3).

Definition 2.1.15 *Für einen ungerichteten Graphen $H = (V_H, E_H)$ mit $|V_H| = n$ heißt eine Nummerierung ϕ perfekt, falls gilt:*

$$N_H(v_j) \cap \{v_1, \ldots, v_{j-1}\} \text{ ist vollständig für alle } j \in \{2, \ldots, n\}.$$

Die gewünschte Charakterisierung der zerlegbaren Graphen liefert der nächste Satz, der eine Folgerung aus Theorem 4.4 und Theorem 4.5 aus dem Buch von Cowell[13] darstellt:

Satz 2.1.16 *Ein ungerichteter Graph H ist zerlegbar genau dann, wenn es eine perfekte Nummerierung gibt.*

Es wird nun der Vorgang der Moralisierung definiert wie in der Arbeit von Dawid und Lauritzen[16]. Die Umkehrung dieses Vorganges wird Prämoralisierung genannt und in Kapitel 3 eingeführt.

Definition 2.1.17 *Sei $G = (V_G, \overrightarrow{E_G})$ ein DAG. Der moralisierte Graph von G, oder auch Moralisierung von G, ist definiert als ein ungerichteter Graph $G^m = (V_{G^m}, E_{G^m})$ mit $V_{G^m} := V_G$ und*

$$E_{G^m} := \{\{v, w\} \in \binom{V_G}{2} \mid (v, w) \in \overrightarrow{E_G} \vee (w, v) \in \overrightarrow{E_G} \vee \exists x \in V_G : v, w \in P_G(x)\}.$$

Man sagt, dass eine Kante $\{v, w\} \in E_{G^m}$ durch Moralisierung entsteht, falls $(v, w) \notin \overrightarrow{E_G}$ und $(w, v) \notin \overrightarrow{E_G}$.

Anschaulich konstruiert man für einen gerichteten Graphen G die Moralisierung dadurch, dass die gerichteten Kanten von G in ungerichtete Kanten umgewandelt werden und zusätzlich alle Eltern eines jeden Knoten paarweise durch eine Kante verbunden werden (siehe Abbildung 1.1).

2.2 Multivariate Normalverteilung

Bei Microarray-Experimenten wird generell davon ausgegangen, dass die erzeugten Daten nach Normalisierung multivariat normalverteilt sind. Auch bei den im nächsten Abschnitt eingeführten Gaußschen graphischen Modellen handelt es sich um eine Klasse von multivariaten Normalverteilungen.

Die Dichte der multivariaten Normalverteilung bezüglich des Lebesguemaßes ist durch zwei Parameter festgelegt, den Mittelwert μ und die Kovarianzmatrix Σ. Da die Kovarianzmatrix bei einer nicht degenerierten multivariaten Normalverteilung, und nur solche werden hier betrachtet, immer positiv definit sein muss, ist es sinnvoll, sich mit den positiv definiten Matrizen zu befassen.

2.2.1 Positiv definite Matrizen

Definition 2.2.1 *Eine Matrix $M = (m_{ij}) \in M(n, n, \mathbb{R})$ heißt positiv definit, falls für jedes $v \in \mathbb{R}^n$ mit $v \neq 0$ gilt:*

$$v^t \cdot M \cdot v > 0.$$

Betrachtet man eine nicht degenerierte multivariate Normalverteilung, so ist die zugehörige Kovarianzmatrix nicht nur positiv definit, sondern auch symmetrisch. Für symmetrische Matrizen ergibt sich folgendes Definitheitskriterium (siehe [23]):

Satz 2.2.2 *Eine symmetrische Matrix $M = (m_{ij}) \in M(n, n, \mathbb{R})$ ist genau dann positiv definit, falls alle Hauptminoren positiv sind. Die Hauptminoren einer Matrix M sind definiert als die Determinanten der Matrizen M_k für $k = 1, \ldots, n$. Hierbei entsteht M_k aus M durch Streichung der Zeilen $k + 1, \ldots, n$ und Spalten $k + 1, \ldots n$.*

Symmetrische positiv definite Matrizen lassen sich einfach aus oberen Dreiecksmatrizen mit vollem Rang erstellen. Denn für eine obere Dreiecksmatrix $L = (l_{ij}) \in M(n, n, \mathbb{R})$ mit $l_{ii} \neq 0$ für alle i gilt, dass $Q = (q_{ij}) \in M(n, n, \mathbb{R})$ definiert als $Q = L^t \cdot L$ symmetrisch und positiv definit ist. Die Symmetrie ergibt sich aus

$$q_{ij} = \sum l_{ki} l_{kj} = q_{ji}.$$

Die Matrix Q ist positiv definit, denn $Q_k = L_k^t \cdot L_k$ und deshalb gilt für alle k

$$\det(Q_k) = \det(L_k^t) \cdot \det(L_k) = \det(L_k)^2 > 0.$$

Somit folgt die Eigenschaft aus Satz 2.2.2. Mit Hilfe des Verfahrens der Cholesky-Zerlegung ist es nun möglich, für eine beliebige symmetrische positiv definite Matrix Q eine obere Dreiecksmatrix L mit $Q = L^t \cdot L$ und $l_{kk} > 0$ zu generieren. Diese Matrix L ist eindeutig, das bedeutet es gilt folgender Satz.

Satz 2.2.3 *Ist* $Q = (q_{ij}) \in M(n, n, \mathbb{R})$ *eine symmetrische positiv definite Matrix, so existiert eine eindeutige obere Dreiecksmatrix* $L = (l_{ij}) \in M(n, n, \mathbb{R})$ *mit*

1. $Q = L^t \cdot L$
2. $l_{kk} > 0$ *für alle* k

Somit erhält man eine bijektive Abbildung zwischen der Menge der symmetrischen, positiv definiten Matrizen, und der Menge der Matrizen der Form $L^t \cdot L$, wobei L eine obere Dreiecksmatrix mit vollem Rang und positiven Diagonalelementen ist. Definiert man die Menge der oberen Dreiecksmatrizen als

$$OD(n) := \{L = (l_{ij}) \in M(n, n, \mathbb{R}) | l_{ij} = 0 \text{ für alle } i, j \text{ mit } j < i\}$$

so ergibt sich aus Satz 2.2.3, dass sich die Menge $SP(n)$ der symmetrischen positiv definiten Matrizen schreiben lässt als

$$SP(n) = \{L \cdot L^t | L \in OD(n) \text{ und } l_{kk} \neq 0 \text{ für alle } k\}.$$

Eine Zerlegung einer symmetrischen positiv definiten Matrix ist in vielen Bereichen sehr sinnvoll. Zum Beispiel lässt sich ein Gleichungssystem der Form $Ax = b$ einfach durch Vorwärts- und Rückwärtseinsetzung lösen, falls eine Zerlegung der Matrix A in $L^t \cdot L$ vorliegt. Zum Erzeugen von Daten aus einer nicht degenerierten multivariaten Normalverteilung ist eine Zerlegung ebenfalls von Nutzen (siehe Kapitel 2.2.3).

2.2.2 Dichte und Korrelationsmatrix

Definition 2.2.4 *Eine Zufallsvariable X auf $\mathbb{R}^n$ besitzt eine nicht degenerierte multivariate Normalverteilung zum Mittelwert $\mu \in \mathbb{R}^n$ und zur Kovarianzmatrix $\Sigma \in SP(n)$, falls die Dichte f_n von X bezüglich des Lebesguemaßes gegeben ist durch*

$$f_n(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\big(-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)\big).$$

Hierbei bezeichnet $|\Sigma|$ die Determinante von Σ. Man schreibt $X \sim N_n(\mu, \Sigma)$, wobei der Index n häufig weggelassen wird.

Es ist auch möglich eine multivariate Normalverteilung zu definieren, falls die Kovarianzmatrix nicht positiv definit, sondern nur positiv semidefinit ist. Eine solche Verteilung wird degenerierte Normalverteilung genannt und besitzt keine Dichte bezüglich des Lebesguemaßes. In dieser Arbeit werden aber nur nicht degenerierte multivariate Normalverteilungen betrachtet, so dass der Term *nicht degeneriert* im Folgenden weggelassen wird.

Für $X \sim N(\mu, \Sigma)$ ist man interessiert an Größen, die den linearen Zusammenhang für je zwei Zufallsgrößen X_i und X_j beschreiben. Dies geschieht mit Hilfe des Korrelationskoeffizienten nach Pearson. Dieser ist allgemein für zwei Zufallsvariablen X und Y definiert durch:

$$Kor(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)} \cdot \sqrt{Var(Y)}}$$

Für X_i und X_j mit $X \sim N(\mu, \Sigma)$ lässt sich die Korrelation aus der Kovarianzmatrix Σ ableiten.

Definition 2.2.5 *Für eine Matrix $A = (a_{ij}) \in M(n,n,\mathbb{R})$ mit $a_{ii} > 0$ für alle i definiert man die Funktion $f : A \to f(A) = B$ mit $B = (b_{ij}) \in M(n,n,\mathbb{R})$ und*

$$b_{ij} := \frac{a_{ij}}{\sqrt{a_{ii}} \cdot \sqrt{a_{jj}}}.$$

Ist $X \sim N(\mu, \Sigma)$ so gilt $Kor(X_i, X_j) = b_{ij}$ mit $B = (b_{ij}) \in M(n,n,\mathbb{R})$ und $f(\Sigma) = B$. B wird als Korrelationsmatrix bezeichnet.

Die Funktion f in Definition 2.2.5 lässt sich darstellen als Produkt der Matrix A mit zwei Diagonalmatrizen. Ist T eine Diagonalmatrix mit $t_{ii} = \frac{1}{\sqrt{a_{ii}}}$, so gilt

$$f(A) = B = T \cdot A \cdot T$$

da

$$b_{ij} = \sum_k \Big(\sum_l t_{il} \cdot a_{lk}\Big) \cdot t_{kj} = t_{ii} \cdot a_{ij} \cdot t_{jj} = \frac{a_{ij}}{\sqrt{a_{ii}} \cdot \sqrt{a_{jj}}}.$$

Variablen, die eine hohe Korrelation aufweisen, müssen nicht notwendigerweise funktionell direkt zusammenhängen. Beispielsweise haben zwei Variablen X und Y eine hohe

Korrelation, falls sie stark von einer dritten Variablen Z abhängig sind. Wirkt eine Zufallsvariable Z sowohl auf eine Variable X als auch auf eine Variable Y, so haben X und Y eine hohe Korrelation. Bezieht man aber den Einfluss von Z mit in die Berechnung ein, das bedeutet, betrachtet man den Einfluss von X auf Y, wenn der Einfluss von Z schon bekannt ist, so wird X keinen zusätzlichen Einfluss mehr auf Y haben. Die Korrelation zwischen Y und X gegeben die Einflüsse aller übrigen Variablen wird als partielle Korrelation bezeichnet. Ähnlich werden auch partielle Varianzen definiert. Beide Begriffe werden in dieser Arbeit nur für eine multivariate Normalverteilung definiert. In diesem Fall lassen sich die partiellen Varianzen und Korrelationen sehr einfach aus der Präzessionsmatrix ableiten, der Inversen der Kovarianzmatrix (siehe [20, 77]).

Definition 2.2.6 *Sei $X \sim N_n(\mu, \Sigma)$. Die bedingte Verteilung von X_i und X_j gegeben $(X_k)_{k\neq i,j}$ ist eine bivariate Normalverteilung, deren Korrelation als partielle Korrelation bezeichnet wird. Die partielle Korrelation zwischen X_i und X_j gegeben $(X_k)_{k\neq i,j}$ kann wie folgt berechnet werden:*

$$Kor(X_i, X_j|(X_k)_{k\neq i,j}) = -\frac{\omega_{ij}}{\sqrt{\omega_{ii}} \cdot \sqrt{\omega_{jj}}}.$$

Hierbei ist $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ die Präzessionsmatrix zu Σ, das bedeutet $\Omega = \Sigma^{-1}$.

Definition 2.2.7 *Sei $X \sim N_n(\mu, \Sigma)$. Die bedingte Verteilung von X_i gegeben $(X_k)_{k\neq i}$ ist eine univariate Normalverteilung, deren Varianz als partielle Varianz bezeichnet wird. Die partielle Varianz von X_i gegeben $(X_k)_{k\neq i}$ kann wie folgt berechnet werden:*

$$Var(X_i|(X_k)_{k\neq i}) = \frac{1}{\omega_{ii}}.$$

Hierbei ist $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ die Präzessionsmatrix zu Σ.

Wendet man die Funktion f aus Definition 2.2.5 auf eine Präzessionsmatrix an, so sind die Nicht-Diagonal-Elemente der resultierenden Matrix die negativen partiellen Korrelationen, das bedeutet, um von der Präzessionsmatrix zu einer Matrix mit den partiellen Korrelationen zu gelangen, muss man nicht nur die Funktion f auf die Präzessionsmatrix anwenden, sondern auch noch für die Nicht-Diagonal-Elemente das Vorzeichen wechseln. Da der Vorgang des Vorzeichenwechsels der Nicht-Diagonal-Elemente in der weiteren Arbeit häufiger benutzt wird, wird dieser jetzt durch eine Funktion g definiert.

Definition 2.2.8 *Für eine Matrix $A = (a_{ij}) \in M(n, n, \mathbb{R})$ definiert man die Funktion $g : A \to g(A) = B$ mit $B = (b_{ij}) \in M(n, n, \mathbb{R})$ und*

$$\begin{aligned} b_{ij} &:= -a_{ij} \text{ für alle } i \neq j \\ b_{ii} &:= a_{ii} \text{ für alle } i. \end{aligned}$$

2.2.3 Simulationstechniken für multivariate Normalverteilungen

Wenn man eine Zerlegung der Präzessionsmatrix ähnlich wie in Satz 2.2.3 besitzt, so ist es sehr einfach, Daten aus einer multivariaten Normalverteilung zu erzeugen. Die Voraussetzungen hierfür liefert der folgende Satz (siehe [40] Prop. C.3).

Satz 2.2.9 *Gilt $x \sim N_p(\mu, \Sigma)$, dann ist $y = Ax+b$ mit $(q\times p)$- Matrix A und $(q\times 1)$-Vektor b normalverteilt mit*

$$y \sim N_q(A \cdot \mu + b, A \cdot \Sigma \cdot A^t).$$

Aus diesem Satz lässt sich ein einfaches Verfahren zur Erzeugung von multivariat verteilten Daten ableiten. Hierzu benötigt man eine Zerlegung der inversen Kovarianzmatrix in das Produkt $K^t \cdot K$, wobei K eine obere Dreiecksmatrix mit vollem Rang ist. Der folgende Algorithmus aus einer Arbeit von Rue[58] erzeugt Daten aus einer multivariaten Normalverteilung $N(0, \Sigma)$, wobei Ω die zu der Normalverteilung gehörende Präzessionsmatrix bezeichnet.

Datensimulation nach Rue

1. Verändere die Matrix Ω durch Permutationen so, dass sie eine kleine Bandbreite $b_w = max_{i\sim j}|i-j|$ besitzt. Es entsteht die permutierte Matrix $\Omega_p = P \cdot \Omega \cdot P^t$.

2. Bilde die Cholesky-Zerlegung der permutierten Matrix $\Omega_p = K^t \cdot K$. Hierbei ist K eine obere Dreiecksmatrix mit vollem Rang.

3. Für z unabhängig standardnormalverteilt löse das Gleichungssystem $K \cdot x_p = z$

4. Wende die Permutation $x = P^t x_p$ an.

Der Vektor x entstammt dann der gewünschten Verteilung mit Mittelwert 0 und Kovarianzmatrix $\Sigma = \Omega^{-1}$. Dies ist eine direkte Folgerung aus nachfolgendem Lemma 2.2.10. Die Permutation P in dem obigen Algorithmus ist ein numerisches Hilfsmittel, um die Cholesky-Zerlegung der Matrix Ω mit geringem Aufwand berechnen zu können. In späteren Anwendungen wird diese Permutation häufig die Identität sein, weil man schon eine passende Zerlegung der Matrix vorliegen hat (siehe Kapitel 4). Diese Zerlegung hat dann auch notwendigerweise keine positiven Diagonalelemente. Dies ist aber auch nicht nötig, es wird im Folgenden nur gefordert, dass die obere Dreiecksmatrix vollen Rang hat.

Lemma 2.2.10 *Sei $z \sim N(0, Id)$ und K eine obere Dreiecksmatrix mit vollem Rang. Sei x die Lösung der Gleichung $K \cdot x = z$. Dann gilt*

$$x \sim N(0, (K^t \cdot K)^{-1}).$$

Beweis: Da K vollen Rang hat, kann man die Matrix invertieren und es ergibt sich $x = K^{-1} \cdot z$. Wendet man nun Satz 2.2.9 an, so erhält man mit $\mu = b = 0$, $\Sigma = Id$ und $A = K^{-1}$, dass

$$x \sim N_p(0, K^{-1} \cdot (K^{-1})^t).$$

Da aber $K^{-1} \cdot (K^{-1})^t = K^{-1} \cdot (K^t)^{-1} = (K^t \cdot K)^{-1}$ ergibt sich die Aussage.

q.e.d.

Mit diesem Lemma lässt sich der folgende Satz beweisen, welcher eine Vorbereitung für Untersuchungen in Kapitel 4 darstellt.

Satz 2.2.11 *Sei* $\Sigma = (\sigma_{ij}) \in M(n, n, \mathbb{R})$ *eine positiv definite symmetrische Matrix und sei* $x \sim N(0, \Sigma)$. *Sei* $\Gamma = (\gamma_{ij}) \in M(n, n, \mathbb{R})$ *eine obere Dreiecksmatrix mit Diagonale* 0 *und* $\Psi = (\psi_{ij}) \in M(n, n, \mathbb{R})$ *eine Diagonalmatrix mit vollem Rang. Falls sich* x *darstellen lässt als*

$$x = \Gamma \cdot x + \epsilon \text{ mit } \epsilon \sim N(0, \Psi)$$

so gilt für die Präzessionsmatrix $\Omega = \Sigma^{-1}$

$$\Omega = (Id_n - \Gamma)^t \cdot \Psi^{-1} \cdot (Id_n - \Gamma).$$

Beweis: Sei $z \sim N(0, Id_n)$. Definiere $B = (b_{ij}) \in M(n, n, \mathbb{R})$ durch

$$\begin{aligned} b_{ij} &= 0 \quad \text{für alle } i, j \\ b_{ii} &= \sqrt{\psi_{ii}} \quad \text{für alle } i \end{aligned}$$

Dann ist B eine Diagonalmatrix mit vollem Rang und es gilt $B \cdot B^t = \Psi$. Somit folgt nach Satz 2.2.9 $B \cdot z \sim N(0, \Psi)$. Also lässt sich x darstellen als

$$x = \Gamma \cdot x + B \cdot z$$

und erfüllt somit die Gleichung

$$K \cdot x = z,$$

wobei $K = (k_{ij}) \in M(n, n, \mathbb{R})$ definiert ist als $K := B^{-1} \cdot (Id_n - \Gamma)$. K ist eine obere Dreiecksmatrix mit vollem Rang. Somit folgt aus Lemma 2.2.10 für die Präzessionsmatrix:

$$\begin{aligned} \Omega &= K^t \cdot K \\ &= (B^{-1} \cdot (Id_n - \Gamma))^t \cdot (B^{-1} \cdot (Id_n - \Gamma)) \\ &= (Id_n - \Gamma)^t \cdot (B \cdot B^t)^{-1} \cdot (Id_n - \Gamma) \\ &= (Id_n - \Gamma)^t \cdot \Psi^{-1} \cdot (Id_n - \Gamma) \end{aligned}$$

q.e.d.

2.3 Gaußsche graphische Modelle

Die Gaußschen graphischen Modelle sind multivariate Normalverteilungen, bei denen zusätzliche Bedingungen an gewisse Elemente der Präzessionsmatrix gestellt sind. Diese zusätzlichen Bedingungen sind durch einen ungerichteten Graphen gegeben.

Definition 2.3.1 *Sei $H = (V, E)$ ein ungericheter Graph mit $|V| = n$. Als Gaußsches graphisches Modell (GGM) auf $\mathbb{R}^n$ induziert durch H wird die Familie von multivariaten Normalverteilungen $N(\mu, \Sigma)$ bezeichnet, wobei μ beliebig ist, und für die Präzessionsmatrix $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ definiert als $\Omega = \Sigma^{-1}$ gilt*

$$\{v, w\} \notin E \Rightarrow \omega_{vw} = 0$$

für alle Knoten $v, w \in V$.

Bei einem GGM werden gewisse Einträge der Präzessionsmatrix als Null vorgegeben. Dies ist gleichbedeutend damit, dass gewisse partielle Korrelationen auf Null gesetzt werden.

Sind für ein GGM die Präzessionsmatrix beziehungsweise die partielle Korrelationsmatrix gegeben, so kann abgelesen werden, welche Kanten mindestens in dem Graphen vorhanden sind, der dem GGM zu Grunde liegt. Es können sich aber durchaus noch mehr Kanten in dem zu Grunde liegenden Graphen befinden, denn für die Einträge an den Stellen, an denen der Graph eine Kante hat, gibt es keine Vorgaben. Das bedeutet, an diesen Stellen kann der Matrixeintrag durchaus Null sein, so dass bestimmte multivariate Normalverteilungen zu verschiedenen GGMs gehören können.

Häufig ist es vorteilhaft, wenn man an die Elemente der Matrix, die zu einer Kante korrespondieren, die zusätzliche Eigenschaft fordert, dass diese Elemente nicht Null sind. In einem solchen Fall wird die Struktur des Graphen durch die Matrix *repräsentiert*.

Definition 2.3.2 *Ein ungerichteter Graph $H = (V, E)$ mit $V = \{1, \ldots, n\}$ wird durch eine symmetrische Matrix $M = (m_{ij}) \in M(n, n, R)$ repräsentiert, falls für alle i,j mit $\{i, j\} \notin E$ gilt, dass $m_{ij} = 0$ und falls für alle i,j mit $\{i, j\} \in E$ gilt, dass $m_{ij} \neq 0$.*

Ein ungerichteter Graph $H = (V, E)$ mit $V = \{1, \ldots, n\}$ wird durch eine symmetrische Matrix $M = (m_{ij}) \in M(n, n, R)$ kanonisch repräsentiert, falls für alle i,j mit $\{i, j\} \notin E$ gilt, dass $m_{ij} = 0$ und falls für alle i,j mit $\{i, j\} \in E$ gilt, dass $m_{ij} = 1$.

Definition 2.3.3 *Ein gerichteter Graph $G = (V, \overrightarrow{E})$ mit $V = \{1, \ldots, n\}$ wird durch eine Matrix $M = (m_{ij}) \in M(n, n, R)$ repräsentiert, falls für alle i,j mit $(i, j) \notin \overrightarrow{E}$ gilt, dass $m_{ji} = 0$, und falls für alle i,j mit $(i, j) \in \overrightarrow{E}$ gilt, dass $m_{ji} \neq 0$.*

Ein gerichteter Graph $G = (V, \overrightarrow{E})$ mit $V = \{1, \ldots, n\}$ wird durch eine Matrix $M = (m_{ij}) \in M(n, n, R)$ kanonisch repräsentiert, falls für alle i,j mit $(i, j) \notin \overrightarrow{E}$ gilt, dass $m_{ji} = 0$, und falls für alle i,j mit $(i, j) \in \overrightarrow{E}$ gilt, dass $m_{ji} = 1$.

Ist der gerichtete Graph G ein DAG, werden die Knoten immer so nummeriert beziehungsweise benannt, dass die repräsentierende Matrix M eine obere Dreiecksmatrix ist. Die Permutation π zwischen einer beliebigen Nummerierung der Knoten in G und einer Nummerierung mit der gewünschten Eigenschaft kann wie folgt definiert werden. Man betrachtet G und die vorhandene Nummerierung der Knoten. Der erste Knoten der neuen Nummerierung ist einer der Knoten v, für die $C_G(v) = \emptyset$ gilt, das heißt der Knoten v hat keine Kinder in G. Mindestens ein solcher Knoten existiert, da G azyklisch ist. Man bildet dann $G_1 = G \backslash v$. Der zweite Knoten der neuen Reihenfolge ist dann einer der Knoten v mit $C_{G_1}(v) = \emptyset$. So wird weiter vorgegangen, bis jeder Knoten eine neue Nummer besitzt. Dieser Vorgang ist natürlich nicht eindeutig, führt aber immer zu einer Nummerierung mit der gewünschten Eigenschaft.

2.4 Algorithmen zur Schätzung eines Netzwerkes aus Genexpressionsdaten

Die folgenden drei Ansätze beschreiben Verfahren, mit denen versucht wird, aus Microarray-Daten Gennetzwerke zu erzeugen, wobei die Kanten partielle Korellationen repräsentieren, die signifikant von Null abweichen. Alle Verfahren resultieren in einem ungerichteten Graphen. Der Ansatz von Schäfer und Strimmer wird vor allem in Kapitel 5 benutzt, um dort Netzwerke aus Microarray-Daten zu schätzen. Der Ansatz von Dobra wird erläutert, weil Teile davon die Grundidee des in Kapitel 4 vorgestellten Simulationsansatzes vermitteln. Der Ansatz von Meinshausen und Bühlmann wird in diese Arbeit aufgenommen, weil bei diesem Ansatz für den Konvergenzfall bewiesen wird, dass die korrekten Resultate geliefert werden.

2.4.1 Ansatz von Schäfer und Strimmer

Schäfer und Strimmer stellen in ihren Arbeiten verschiedene Algorithmen vor, mit deren Hilfe man die partielle Korrelationsmatrix schätzen und dann testen kann, welche Einträge sich signifikant von Null unterscheiden (siehe [63, 64]). Der in Kapitel 5 benutzte Algorithmus wird in [64] vorgestellt. Hier wird ein *Shrinkage*-Ansatz in Kombination mit dem Ledoit-Wolf Lemma[41] benutzt, um die Kovarianzmatrix und schließlich die partielle Korrelationsmatrix zu schätzen. Das bedeutet, der Schätzer S der Kovarianzmatrix ergibt sich aus

$$S = \lambda \cdot T + (1 - \lambda) \cdot U$$

wobei T eine Diagonalmatrix und U der empirische Kovarianzschätzer ist. Der optimale Wert für λ wird über das Ledoit-Wolf Lemma ermittelt. Für den Test auf Signifikanz wird angenommen, dass die geschätzten partiellen Korrelationen einer Mischverteilung entstammen. Das bedeutet, die Verteilung ist eine gewichtete Mischung zwischen der Verteilung der partiellen Korrelationen für vorhandene Kanten und der Verteilung, falls keine partielle

Korrelation vorliegt, der Nullverteilung. Letztere kann nach Hotelling[36] berechnet werden. Mit diesen Annahmen lässt sich für jeden Eintrag eine *local false discovery rate*[21] berechnen, man erhält also für jeden Eintrag eine Wahrscheinlichkeit dafür, dass sich signifikant von Null unterscheidet.

Das Resultat bei Schäfer und Strimmer ist ein ungerichteter Graph, der gerade die Kanten enthält, bei denen sich der Eintrag in der geschätzten partiellen Korrelationsmatrix signifikant von Null unterscheidet. Die Signifikanzschwelle, in dieser Arbeit auch als q-Wert-Schranke bezeichnet, wird in den Schäfer/Strimmer-Arbeiten und in dieser Arbeit häufig auf 0.2 gesetzt.

Eine Schwierigkeit des Ansatzes betrifft die Definitheit. Ist S die geschätzte Kovarianzmatrix, so bezeichne $K := g(f(S^{-1}))$ die daraus resultierende partielle Korrelationsmatrix. Hierbei entsprechen f und g den Funktionen aus Definition 2.2.5 und Definition 2.2.8. Diese Matrix ist wieder positiv definit. Erzeugt man aus K die Matrix $\bar{K}$, indem man nur die Einträge behält, die signifikant unterschiedlich von Null sind, und alle anderen Einträge auf Null setzt, so würde man gerne diese Matrix als die partielle Korrelationsmatrix der Verteilung ansehen, aus der die Genexpression entstammt. Dies ist aber nicht allgemein möglich, da die Matrix $\bar{K}$ nicht notwendigerweise positiv definit ist (siehe Bemerkungen am Anfang von Kapitel 4). Dieses Problem kann beispielsweise beim parametrischen Bootstrap in Kapitel 5.4 auftreten.

2.4.2 Ansatz von Dobra

In der Arbeit von Dobra[18] wird über ein mehrstufiges Verfahren versucht, einen ungerichteten Graphen zu erzeugen, der partielle Korrelationen repräsentiert. In einem ersten Schritt werden für jedes Gen andere Gene bestimmt, die die Expression des Gens am besten vorhersagen. Im Detail wird für die Expression jedes Gens eine lineare Regression durchgeführt, wobei zuerst die Expressionen aller anderen Gene als erklärende Variablen eingehen. Dann wird ein für jedes Gen möglichst gutes Modell gefunden, wobei hierbei Vorwärts- beziehungsweise Rückwärtssuche benutzt wird. Die finalen Modelle sollen zusätzlich nur wenig erklärende Variablen besitzen. So wird für jedes Gen i eine Menge $pv(i)$ gefunden, deren Elemente die beste Vorhersagekraft für die Expression von Gen i besitzen.

Im zweiten Schritt wird eine Umordnung der Gene vorgenommen. Hierzu wird eine Ordnung auf der Menge der Gene eingeführt, die angibt, wie wichtig jedes einzelne Gen bei der Vorhersage der anderen Gene ist. Die Wichtigkeit ergibt sich aus den linearen Gleichungen aus Schritt 1. Das unwichtigste Gen wird herausgenommen und notiert. Es ist das erste Gen in der neuen Ordnung, denn es ist zur Beschreibung der anderen Gene nicht sehr wichtig. Dann wird der erste Schritt ein weiteres Mal durchgeführt, wobei das oben notierte Gen nicht mehr benutzt werden darf. Dieses Verfahren wird iterativ fortgesetzt, bis jedes Gen eine neue Nummer besitzt.

Dobra konstruiert durch dieses Vorgehen ein Gleichungssystem, um eine Dichte p zu beschreiben, die Dichte einer multivariaten Normalverteilung mit beliebigem Mittelwert

und mit Präzessionsmatrix Ω. Die Dichte hat die Form

$$p(x) = \prod_{i=1}^{p-1} p(x_i | x_{cpv(i)}) p(x_p).$$

Hierbei gilt $cpv(i) \subseteq \{(i+1) : p\}$ wobei die oben erstellte Reihenfolge der Gene, also der Variablen, benutzt wird. Die Form der Dichte p impliziert ein Gleichungssystem

$$x = \Gamma x + \epsilon \qquad \epsilon \sim N_p(0, \Psi)$$

wobei Ψ eine Diagonalmatrix ist und Γ eine obere Dreiecksmatrix mit $\Gamma_{ii} = 0$ für alle i.

Zu diesem Gleichungssystem konstruiert Dobra einen gerichteten azyklischen Graphen $G = (V, \overrightarrow{E})$, so dass die Dichte p über G rekursiv faktorisiert (siehe Kapitel 4.2). Das geschieht, indem man gerichtete Kanten von jedem Element der Menge $cpv(i)$ zum Element i einfügt, das heißt

$$(j, i) \in \overrightarrow{E} \Leftrightarrow j \in cpv(i).$$

Der so erzeugte Graph wird moralisiert. Es entsteht der Graph $G^m = (V, E^m)$. Dieser Graph wird als Visualisierung der Präzessionsmatrix angesehen, denn nach Dobra soll gelten

$$\Omega_{vw} \neq 0 \Leftrightarrow \{v, w\} \in E^m.$$

Die Hinrichtung dieser Aussage gilt tatsächlich und ist eine Folgerung aus Satz 4.2.2. Diese Aussage ist auch die Kernidee des in Kapitel 4 vorgestellten Simulationsalgorithmus. Wie aber in Kapitel 4 gezeigt wird, ist die Rückrichtung der Aussage im Allgemeinen nicht gültig. Betrachtet man also nur den bei Dobra erzeugten ungerichteten Graphen, so ist dies keine Repräsentation der von Dobra geschätzten Präzessionsmatrix. Es können Kanten in dem moralisierten Graphen G^m auftreten, obwohl der geschätzte Eintrag in der Präzessionsmatrix Null ist. Dies ist natürlich ein Nachteil des Ansatzes, falls man den moralisierten Graphen als Resultat ansieht. Das Problem ergibt sich nicht, falls man nur an der erzeugten positiv definiten Präzessionsmatrix interessiert ist, die man ebenfalls mit dem Ansatz berechnen kann. Die beschriebene Inkonsistenz zwischen den betrachteten Matrizen und der Moralisierung eines Graphen war auch ein Anstoß für die vorliegende Arbeit, um Zusammenhänge zwischen einer Moralisierung eines Graphen und der den Graphen repräsentierenden Matrizen genauer zu untersuchen.

2.4.3 Ansatz von Meinshausen und Bühlmann

Auch Meinshausen und Bühlmann [50] setzen in ihrer Arbeit voraus, dass die vorliegenden Daten $X = (X_1, \ldots, X_p)$ einer multivariaten Normalverteilung $N(\mu, \Sigma)$ entstammen. Sie möchten einen Graphen H mit p Knoten erstellen, bei dem jeder Knoten v mit allen anderen Knoten w verbunden ist, für die der entsprechende Eintrag in der zu Σ gehörenden Präzessionsmatrix $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ nicht Null ist. Ziel ist es also für jeden beliebigen Knoten v die Menge $N_H(v)$ zu bestimmen. Die Einträge der Präzessionsmatrix haben einen

engen Zusammenhang mit den Koeffizienten, die sich bei der Regression der Variable X_v bezüglich aller übrigen Variablen ergeben. Betrachtet man

$$\theta^v = \arg\min_{\theta:\theta_v=0} E(X_v - \sum_{w\in H} \theta_w X_w)^2$$

so gilt $\omega_{vw} = 0 \Leftrightarrow \theta^v_w = 0$, also kann man die Frage nach den Nachbarn auch beantworten, indem man θ^v untersucht. Meinshausen und Bühlmann benutzen die Lasso-Methode[70], um θ^v und somit $N_H(v)$ abzuschätzen. Der Lasso-Schätzer für θ^v ist gegeben durch

$$\hat{\theta}^{v,\lambda} = \arg\min_{\theta:\theta_v=0} (n^{-1}\|X_v - X\theta\|_2^2 + \lambda\|\theta\|_1).$$

und hängt sehr stark vom Bestrafungsterm λ ab.

Benutzt man diesen Ansatz, um alle Nachbarschaften der einzelnen Knoten zu bestimmten, so können Inkonsistenzen auftreten, beispielsweise kann es sein, dass $\hat{\theta}^{v,\lambda}_w \neq 0$ aber $\hat{\theta}^{w,\lambda}_v = 0$, und damit wäre w ein Nachbar von v aber v kein Nachbar von w. In ihrer Arbeit zeigen die Autoren, dass dieser Fall asymptotisch unter gewissen Nebenbedingungen nicht mehr auftreten kann. Es wird die stärkere Aussage bewiesen, dass für n gegen unendlich die Wahrscheinlichkeit, dass die gefundene Nachbarschaft mit der wirklichen Nachbarschaft übereinstimmt, gegen 1 konvergiert. n ist hierbei die Anzahl der in die Untersuchungen einfließenden Beobachtungen. Von n hängt dann auch die Anzahl der eingehenden Parameter p, also die Anzahl der Knoten in den Graphen, die Verteilung, also Σ, und auch der Bestrafungsterm λ ab.

Zusätzlich zum asymptotischen Verhalten wird in dem Paper auch gezeigt, dass man die Wahrscheinlichkeit, durch die Schätzung zwei Zusammenhangskomponenten zu verbinden, bei einer geschickten Wahl des Bestrafungsterms λ beschränken kann. Dies ist vor allem für die Anwendung des Algorithmus nützlich.

2.5 Optimierungsverfahren

In Kapitel 4 werden zwei verschiedene Verfahren benutzt, um eine dort definierte Funktion zu optimieren. Im folgenden Abschnitt werden die Verfahren kurz vorgestellt. Die Verfahren wurden ausgewählt, weil es sehr elementare Verfahren sind und weil Implementierungen in der Programmiersprache R[37] vorliegen. Somit sind sie leicht zugängig.

2.5.1 Nelder-Mead-Verfahren

Nelder und Mead[51] benutzen einen Simplexalgorithmus, um ein lokales Minimum einer gegebenen Funktion zu erhalten. Hierbei wird der schlechteste Punkt des Simplexes durch eine neue Punktschätzung ausgetauscht. Dieses Verfahren lässt sich am besten für eine zwei-dimensionale Funktion $f : \mathbb{R}^2 \to \mathbb{R}$ verdeutlichen. In diesem Fall besteht ein Simplex aus einem Dreieck. In einem Schritt des Algorithmus wird der Funktionswert von f für die drei Punkte des Dreiecks berechnet. Sei A der Punkt mit dem kleinsten Funktionswert und

C der Punkt mit dem größten Funktionswert. Der Punkt C soll durch einen neuen Punkt ausgetauscht werden, wobei sich die potentiellen Kandidaten auf der Geraden befinden, die durch C und dem Mittelpunkt zwischen A und B verläuft.

Findet sich auf dieser Geraden kein Punkt mit kleinerem Funktionswert, so wird C und B geändert. C wird ersetzt durch den Mittelpunkt von A und C, sowie B durch den Mittelpunkt von A und B. Der Algorithmus wird fortgesetzt, bis er in einem lokalen Minimum endet.

2.5.2 BFGS-Verfahren

Die Broyden-Fletscher-Goldfarb-Shannon Methode (BFGS-Methode)[9, 24, 30, 66] ist ein spezielles *Quasi-Newton-Verfahren*. Wie beim Newton-Verfahren wird beim Quasi-Newton-Verfahren eine Folge $(x_k)_k$ generiert, die gegen ein lokales Maximum oder Minimum der zu untersuchenden Funktion f konvergiert. Beim Newton-Verfahren wird hierzu die Funktion f im Punkt x_k mit Hilfe der Taylor-Entwicklung durch eine quadratische Funktion approximiert. Der nächste Punkt x_{k+1} wird bestimmt mit Hilfe der Ableitung dieser quadratischen Funktion. Hierzu benötigt man die Inverse der Hesse-Matrix von f im Punkt x_k und den Gradienten von f. Bei einem Quasi-Newton-Verfahren wird, im Gegensatz zum Newton-Verfahren, nicht die Inverse der Hesse-Matrix im Punkt x_k benutzt, sondern eine Approximation H_k. Dieses H_k wird mit jedem Schritt durch ein rechentechnisch effizientes Additionsverfahren neu berechnet.

Im Detail wird zuerst für den Punkt x_k der Nachfolgepunkt x_{k+1} bestimmt, wobei man die Richtung, in der sich dieser Punkt befindet, so wählt, dass die quadratische Approximation minimiert wird. Es ergibt sich

$$x_{k+1} = x_k + \alpha_k \cdot d_k \quad \text{mit} \quad d_k = -H_k^{-1} \cdot \nabla f(x_k)$$

für ein optimal zu wählendes α_k. Bei der BFGS-Methode ergibt sich H_{k+1} aus H_k durch

$$H_{k+1} = H_k + \frac{y_k \cdot y_k^t}{y_k^t \cdot s_k} - \frac{H_k \cdot s_k \cdot s_k^t \cdot H_k^t}{s_k^t \cdot H_k \cdot s_k}$$

für $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$ und $s_k = \alpha_k \cdot d_k$. Die erzeugten Matrizen H_k erfüllen verschiedene geforderte Bedingungen, insbesondere sind sie symmetrisch und positiv definit, so dass alle Rechenschritte wohldefiniert sind.

Kapitel 3

Prämoralisierung eines Graphen

In diesem Kapitel werden gerichtete azyklische Graphen und deren Moralisierungen betrachtet. Speziell betrachtet man die Umkehrung des Moralisierungsvorganges, das bedeutet man möchte für einen gegebenen ungerichteten Graphen H einen DAG G finden, so dass die Moralisierung von G gerade H ergibt. Diese Umkehrung ist bis jetzt in der Literatur nicht detailliert betrachtet worden. Nur in [47] wird ein Zusammenhang zwischen prämoralisierbaren Graphen (Definition siehe unten) und zerlegbaren Graphen erwähnt, der in dieser Arbeit auch bewiesen wird (Theorem 3.1.7). Alle weiteren in diesem Kapitel vorgestellten Lemmata, Sätze und Beweise sind neu und existieren nach Wissen des Autors nicht in der Literatur. Sie bilden die theoretische Grundlage für die weiteren Untersuchungen. Insbesondere die Umkehrung der Moralisierung liefert einen wichtigen Baustein in dem in Kapitel 4 vorgestellten Algorithmus zur Erstellung von positiv definiten Matrizen mit Nebenbedingungen.

3.1 Definition und Eigenschaften

In diesem Abschnitt wird die Prämoralisierung eines Graphen definiert und erste Eigenschaften werden gezeigt. Es ergibt sich, dass nicht jeder Graph prämoralisierbar ist und zudem eine Prämoralisierung nicht eindeutig ist. Weiterhin wird gezeigt, dass die Menge der zerlegbaren Graphen echt enthalten ist in der Menge der prämoralisierbaren Graphen.

Definition 3.1.1 *Sei H ein ungerichteter Graph. Sei G ein DAG mit $G^m = H$. Dann heißt G ein prämoralisierter Graph zu H oder auch Prämoralisierung von H. Ein Graph H heißt prämoralisierbar, falls ein zu H prämoralisierter Graph G existiert.*

Nicht jeder Graph H ist prämoralisierbar. Das kleinste Gegenbeispiel ist ein einfacher Zykel der Länge 4 (Abbildung 3.1, A). Dass ein solcher Zykel nicht prämoralisierbar ist, ist der Inhalt des Satzes 3.1.3. Zur Vorbereitung benötigt man noch ein Lemma.

Lemma 3.1.2 *Sei $H = (V_H, E_H)$ ein prämoralisierbarer Graph und $G = (V_G, \overrightarrow{E_G})$ eine Prämoralisierung von H. Sei $\{v, w\} \in E_H$. Wenn v und w keinen gemeinsamen Nachbarn in H haben, so gilt $(v, w) \in \overrightarrow{E_G}$ oder $(w, v) \in \overrightarrow{E_G}$.*

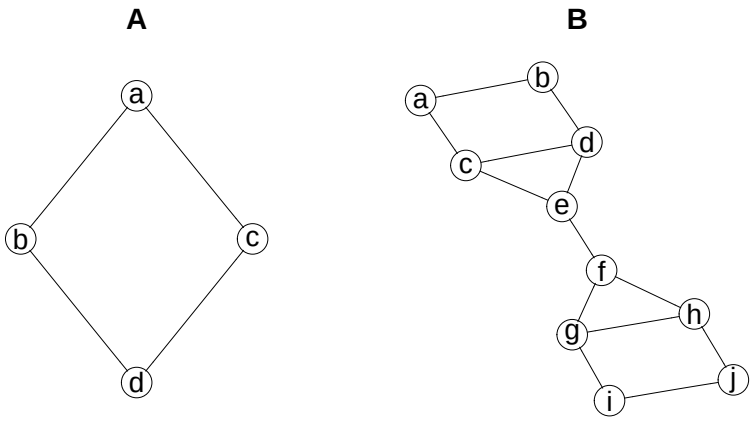

Abbildung 3.1: Zwei nicht prämoralisierbare Graphen: A) Zykel der Länge 4 ohne Abkürzung; B) Ein nicht prämoralisierbarer Graph, bei dem für jeden Zykel ohne Abkürzung zwei im Zykel benachbarte Knoten mit gemeinsamen Nachbarn existieren.

Beweis: Es wird die Negation der Aussage bewiesen. Seien also $(v, w) \notin \overrightarrow{E_G}$ und $(w, v) \notin \overrightarrow{E_G}$. Da aber $\{v, w\} \in E_H$ muss $\{v, w\}$ durch Moralisierung entstanden sein. Also existiert ein $x \in V_G$ mit $v, w \in P_G(x)$. Daraus folgt dann $(v, x), (w, x) \in \overrightarrow{E_G}$ und somit $\{v, x\}, \{w, x\} \in E_H$. Dann ist x ein gemeinsamer Nachbar von v und w.

q.e.d.

Satz 3.1.3 *Sei $Z = v_0, v_1, \ldots, v_{n-1}, v_0$ ein Zykel ohne Abkürzung mit $n \geq 4$. Dann ist Z nicht prämoralisierbar.*

Beweis: Angenommen Z ist doch prämoralisierbar. Dann existiert ein DAG $G = (V_G, \overrightarrow{E_G})$ mit $G^m = Z$. Da der Zykel keine Abkürzung hat und somit im Zykel benachbarte Knoten keinen gemeinsamen Nachbarn haben, gilt nach Lemma 3.1.2, dass für alle im Zykel benachbarte Knoten v_i und v_j entweder $(v_i, v_j) \in \overrightarrow{E_G}$ oder $(v_j, v_i) \in \overrightarrow{E_G}$. Es dürfen aber auch nicht zwei gerichtetete Kanten auf den selben Knoten zeigen, das bedeutet es existiert kein $i \in \{0, \ldots, n-1\}$ mit

$$(v_i, v_{(i+1) \bmod n}), (v_{(i+2) \bmod n}, v_{(i+1) \bmod n}) \in \overrightarrow{E_G},$$

denn dann würde eine zusätzliche Kante durch Moralisierung entstehen, also würde gelten

$$\{v_{(i+2) \bmod n}, v_i\} \in G^m = Z,$$

und somit hätte Z eine Abkürzung. Also gilt für alle i, dass

$$(v_i, v_{(i+1) \bmod n}), (v_{(i+1) \bmod n}, v_{(i+2) \bmod n}) \in \overrightarrow{E_G}$$

aber dann ist G ein Zykel und das ist ein Widerspruch zur DAG Eigenschaft.

q.e.d.

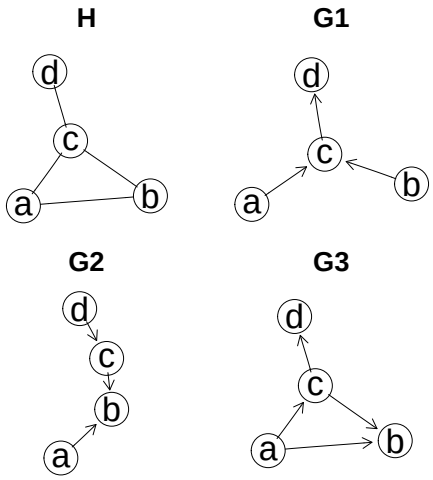

Abbildung 3.2: Beispiel eines ungerichteten Graphens H mit drei möglichen Prämoralisierungen.

Man ist an weiteren Charakterisierungen und Eigenschaften der prämoralisierbaren Graphen interessiert. Betrachtet man die Moralisierung als Abbildung, so ist diese nicht injektiv, das bedeutet zu einem gegebenen Graphen H kann es zwei verschiedene Prämoralisierungen $G1$ und $G2$ geben. Ein einfaches Beispiel zeigt Abbildung 3.2. Für die spätere Anwendung von prämoralisierbaren Graphen in dem Simulationsalgorithmus (Kapitel 4) ist dies aber kein Nachteil. Da man schon gesehen hat, dass nicht jeder Graph prämoralisierbar ist, diese Eigenschaft jedoch für den Simulationsansatz in Kapitel 4 notwendig ist, ist man an notwendigen oder hinreichenden Kriterien interessiert, mit deren Hilfe man bestimmen kann, ob ein Graph prämoralisierbar ist oder nicht.

Durch die Moralisierung eines gerichteten azyklischen Graphen G werden für einen Knoten v Kanten zwischen den Eltern von v in dem resultierenden Graphen H eingefügt. Somit entstehen in H zwischen den Nachbarknoten von v vollständige Knotenmengen. Hat ein Knoten v in G keine Kinder, so werden nach Moralisierung die Nachbarknoten einen vollständigen Teilgraphen bilden. Deshalb ist folgende Definition hilfreich für das weitere Vorgehen.

Definition 3.1.4 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph. Dann heißt ein Knoten $v \in V_H$ voll in H, falls $N_H(v) = \emptyset$ oder $N_H(v)$ in H eine vollständige Knotenmenge bildet.*

Da in einem gerichteten Graphen immer ein Knoten ohne Kinder existiert, ergibt sich mit obiger Definition ein erstes notwendiges Kriterium.

Satz 3.1.5 *In jedem prämoralisierbaren Graphen H gibt es einen vollen Knoten.*

Beweis: Sei $H = (V, E_H)$ ein prämoralisierbarer Graph und sei $G = (V, \overrightarrow{E_G})$ ein gerichteter azyklischer Graph G mit $G^m = H$, also eine Prämoralisierung von H. Sei

$|V| = n$. Da G azyklisch ist, existiert eine Nummerierung ϕ der Knoten, so dass Kanten nur von Knoten mit größeren Labeln zu Knoten mit kleineren Labeln verlaufen. Für alle i sei $v_i = \phi^{-1}(i)$ der Knoten mit dem Label i. Dann gilt für alle $(v_i, v_j) \in \overrightarrow{E_G}$ dass $j < i$ (siehe auch Kapitel 2.3). Man betrachte nun den Knoten v_1 mit dem kleinsten Label. v_1 ist nicht Elternteil eines weiteren Knoten in G. Das bedeutet aber, dass es für ein beliebiges i keine Kante $\{v_i, v_1\} \in E_H$ gibt, die durch Moralisierung entsteht, denn dazu müsste der Knoten v_1 noch Elternteil eines anderen Knotens sein. Somit gilt $(v_i, v_1) \in \overrightarrow{E_G} \iff \{v_i, v_1\} \in E_H$. Dann entsprechen die Nachbarn von v_1 im Graphen H den Eltern im prämoralisierten Graphen G und müssen somit nach Definition der Moralisierung eine vollständige Menge in H bilden. Somit ist v_1 ein voller Knoten in H.

q.e.d.

Zykel ohne Abkürzung sind wie in Satz 3.1.3 gezeigt nicht prämoralisierbar. Vereinfacht gesprochen liegt das Problem für diese Graphen darin, dass man die gerichteten Kanten entweder so einfügt, dass man einen gerichteten Zykel bekommt, oder aber man erhält eine zusätzliche Kante durch Moralisierung. Enthält nun ein ungerichter Graph H einen Zykel ohne Abkürzung als einen Teilgraphen, so muss dieser Graph weitere Kriterien erfüllen, um prämoralisierbar zu sein. Für mindestens eine Kante des Zykels muss es möglich sein, dass diese durch Moralisierung entsteht, denn die übrigen Kanten kann man dann so anordnen, dass man einerseits keine gerichteten Zykel enthält und dass andererseits auch keine unerwünschte Kante durch Moralisierung hinzu kommt. Dieses notwendige Kriterium ist in Satz 3.1.6 formuliert.

Satz 3.1.6 *Sei $H = (V, E)$ ein prämoralisierbarer Graph und sei $Z = v_0, v_1, \ldots, v_{n-1}, v_0$ mit $n \geq 4$ ein Zykel ohne Abkürzung und Teilgraph von H. Dann gibt es zwei im Zykel benachbarte Knoten v_i und $v_{(i+1) \bmod n}$ mit einem gemeinsamen Nachbarn.*

Beweis: Der Beweis verläuft analog zu dem Beweis in Lemma 3.1.3. Es wird die Umkehrung der Aussage gezeigt. Sei also $H = (V, E)$ ein Graph, welcher einen Zykel $Z = v_0, v_1, \ldots, v_{n-1}, v_0$ ohne Abkürzung als Teilgraph enthält, bei dem es keine zwei aufeinanderfolgenden Knoten gibt, die einen gemeinsamen Nachbarn haben. Man muss zeigen, dass H nicht prämoralisierbar ist. Angenommen H ist prämoralisierbar, und $G = (V_G, \overrightarrow{E_G})$ sei eine Prämoralisierung. Nach Lemma 3.1.2 gilt dann, dass für jedes $i \in \{0, \ldots, n-1\}$ entweder $(v_i, v_{(i+1) \bmod n}) \in \overrightarrow{E_G}$ oder $(v_{(i+1) \bmod n}, v_i) \in \overrightarrow{E_G}$. Also sind alle Kanten des Zykels auch in der Prämoralisierung vorhanden. Da G keinen gerichteten Zykel enthält, existiert ein i mit

$$(v_i, v_{(i+1) \bmod n}), (v_{(i+2) \bmod n}, v_{(i+1) \bmod n}) \in \overrightarrow{E_G}.$$

Aber dann entsteht die Kante $\{v_i, v_{(i+2) \bmod n}\}$ durch Moralisierung und somit hat der Zykel eine Abkürzung. Das ist aber ein Widerspruch.

q.e.d.

Die Umkehrung des obigen Satzes ist nicht korrekt. Es gilt nicht, dass ein Graph prämoralisierbar ist, falls es für jeden Zykel ohne Abkürzung $Z = v_0, \ldots, v_{n-1}, v_0$ in diesem

Graphen ein $i \in \{0, \ldots, n-1\}$ gibt, so dass v_i und $v_{(i+1) \bmod n}$ einen gemeinsamen Nachbarn haben. Man betrachte dazu das Beispiel B aus Abbildung 3.1. Nach Satz 3.1.5 ist der Graph nicht prämoralisierbar, denn es gibt keinen vollen Knoten, aber für jeden Zykel ohne Abkürzung gibt es zwei im Zykel benachbarte Knoten mit gemeinsamen Nachbarn.

Die Zykel ohne Abkürzung sind dafür verantwortlich, dass ein Graph H unter Umständen nicht prämoralisierbar ist. Besitzt ein Graph keine Zykel ohne Abkürzung, ist der Graph also zerlegbar, so ist dieser Graph prämoralisierbar. Dies lässt sich mit Hilfe der perfekten Nummerierung zeigen, die man für eine zerlegbaren Graphen hat. Insgesamt erhält man somit also ein hinreichendes Kriterium dafür, dass ein Graph prämoralisierbar ist. Dies ist der Inhalt des folgenden Satzes.

Theorem 3.1.7 *Für jeden zerlegbaren Graphen H gibt es einen zu H prämoralisierten Graphen G.*

Beweis: Sei $H = (V_H, E_H)$ ein zerlegbarer Graph. Dann existiert nach Satz 2.1.16 eine perfekte Nummerierung ϕ. Für alle i sei $v_i = \phi^{-1}(i)$ der Knoten mit dem Label i. Man erstellt aus dem Graphen H den gerichteten Graphen G, indem man die ungerichteten Kanten durch gerichtete ersetzt, wobei die Richtung immer von dem Knoten mit dem kleineren Label zu dem Knoten mit dem größeren Label verläuft. Sei also $G = (V_G, \overrightarrow{E_G})$ mit $V_G = V_H$ und $\overrightarrow{E_G} := \{(v_i, v_j) \mid \{v_i, v_j\} \in E_H \wedge i < j\}$. Der gerichtete Graph G ist ein DAG, da es keine Kante (v_i, v_j) gibt mit $j < i$, also kann es keine Zykel geben.
Zu zeigen ist $G^m = H$. Da $V_{G^m} = V_G = V_H$ bleibt zu zeigen $E_{G^m} = E_H$. Nach Konstruktion von G gilt $E_H \subseteq E_{G^m}$. Sei nun $\{v_i, v_j\} \in E_{G^m}$ mit $i < j$. Nach Definition existiert nun entweder die gerichtete Kante (v_i, v_j) in G, also auch $\{v_i, v_j\}$ in H, oder die Kante ist durch Moralisierung entstanden, das heißt es existiert ein Knoten v_k in G mit $(v_i, v_k) \in \overrightarrow{E_G}$ und $(v_j, v_k) \in \overrightarrow{E_G}$. Dann gilt aber $k > j > i$. In H existieren dann die Kanten $\{v_i, v_k\}$ und $\{v_j, v_k\}$, das heißt $v_i, v_j \in N_H(v_k)$. Nach Definition einer perfekten Nummerierung ist aber $N_H(v_k) \cap \{v_1, \ldots, v_{k-1}\}$ vollständig, also gilt insbesondere $\{v_i, v_j\} \in E_H$.

q.e.d.

Die Menge der zerlegbaren Graphen ist echt kleiner als die Menge der prämoralisierbaren Graphen. Abbildung 3.3 ist ein Beispiel für einen Graphen, der nicht zerlegbar, aber prämoralisierbar ist. Zusammengenommen ergibt sich somit folgende Einordnung der prämoralisierbaren Graphen:

zerlegbare Graphen $\subset$ prämoralisierbare Graphen $\subset$ ungerichtete Graphen

Da die Klasse der zerlegbaren Graphen echt enthalten ist in der Menge der prämoralisierbaren Graphen, ergibt die Zerlegbarkeit auch kein notwendiges *und* hinreichendes Kriterium für die Prämoralisierungseigenschaft. Wie stark sich prämoralisierbare und zerlegbare Graphen unterscheiden, ist nicht bekannt. In Kapitel 5 wird eine Studie durchgeführt, die untersucht, wie groß der Anteil der prämoralisierbaren aber nicht zerlegbaren Graphen ist, wenn Graphen aus Microarray-Daten geschätzt werden.

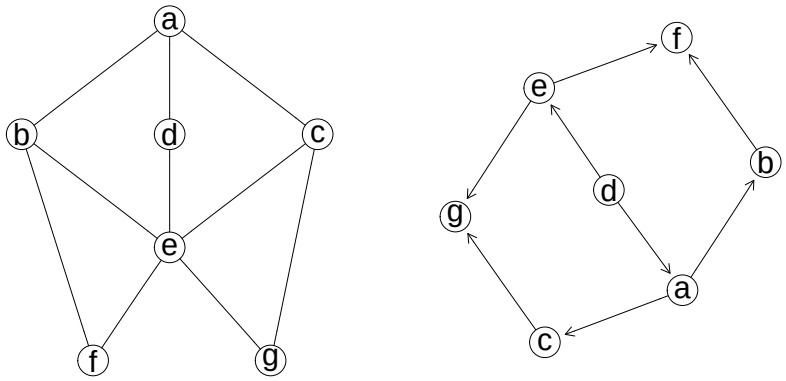

Abbildung 3.3: Ein nicht zerlegbarer Graph und eine zugehörige Prämoralisierung.

3.2 Algorithmen zur Prämoralisierung

Im letzten Abschnitt wurden zwei notwendige Kriterien und ein hinreichendes Kriterium für prämoralisierbare Graphen bewiesen. Allerdings ist noch kein notwendiges und zugleich hinreichendes Kriterium beschrieben worden. Es ist also bis jetzt nicht möglich zu erkennen, ob ein Graph prämoralisierbar ist oder nicht. Zudem wurde noch nicht gezeigt, wie es möglich ist, zu einem gegebenen prämoralisierbaren Graphen eine Prämoralisierung zu finden. In diesem Abschnitt werden nun zwei Algorithmen vorgestellt, die für einen prämoralisierbaren Graphen eine Prämoralisierung finden. Der zweite Algorithmus liefert dann zusätzlich noch das gewünschte notwendige und hinreichende Kriterium. Der erste Algorithmus ist nur für zerlegbare Graphen nutzbar. Der zweite Algorithmus gilt für jeden beliebigen Graphen, ist allerdings auch sehr viel rechenintensiver als Algorithmus I. Den größten Teil des Abschnittes nimmt schließlich der Beweis ein, dass Algorithmus II funktioniert.

3.2.1 Algorithmus für zerlegbare Graphen

Der Beweis des Satzes 3.1.7 impliziert einen Algorithmus, mit dem man für einen zerlegbaren Graphen eine Prämoralisierung findet. Dieser Algorithmus setzt sich zusammen aus der im Beweis von Theorem 3.1.7 angewandten Konstruktion und einem zusätzlichen Kanten-Reduktionsschritt. Zusammengefasst ergibt sich folgender Algorithmus für einen zerlegbaren Graphen H:

Algorithmus I: Prämoralisierungsalgorithmus für zerlegbare Graphen

1. Erstelle für H eine perfekte Nummerierung und bezeichne die Knoten entsprechend dieser Nummerierung.

2. Erzeuge aus H einen gerichteten Graphen G^*, so dass die Kanten von Knoten mit kleinerem Label zu Knoten mit größerem Label verlaufen (siehe Beweis zu Theorem 3.1.7), das heißt für $i < j$ gilt $\{v_i, v_j\} \in E_H \Leftrightarrow (v_i, v_j) \in \overrightarrow{E_{G^*}}$.

3. Betrachte den Graphen G^*. Beginne bei dem Knoten mit dem größten Label und entferne Kanten, die sich zwischen den Eltern dieses Knotens befinden.

4. Wiederhole Schritt 3, bis der Knoten mit dem kleinsten Label erreicht ist.

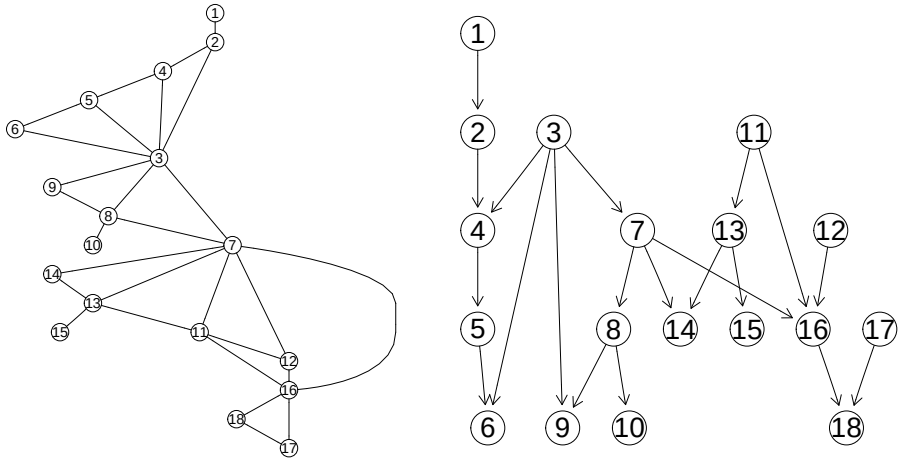

Abbildung 3.4: Beispiel von einem zerlegbaren Graphen und einer dazu passenden, durch Algorithmus I erzeugten Prämoralisierung. Die Knotenlabel entsprechen den Nummern der benutzten perfekten Nummerierung.

Abbildung 3.4 zeigt ein Beispiel für die Anwendung des Algorithmus. Es gilt nun:

Satz 3.2.1 *Sei H ein zerlegbarer Graph. Wendet man Algorithmus I auf H an, so erhält man einen zu H prämoralisierten Graphen G.*

Beweis: Nach Theorem 3.1.7 ist der im zweiten Schritt des Algorithmus I entstehende Graph G^* eine Prämoralisierung von H. Dieser Graph ist insbesondere ein DAG, und aus diesem Grund ist der durch den Algorithmus erzeugte gerichtete Graph G auch ein DAG, denn durch Entfernen von Kanten bleibt ein azyklischer Graph azyklisch. Da nach Theorem 3.1.7 $(G^*)^m = H$ muss man nun noch zeigen, dass $(G^*)^m = G^m$. Nach Konstruktion gilt aber $E_G \subseteq E_{G^*}$ und somit $G^m \subseteq (G^*)^m$. Deshalb bleibt zu zeigen, dass $(G^*)^m \subseteq G^m$.

Sei $\{v_i, v_j\} \in (G^*)^m$ mit $i < j$. Es muss gezeigt werden, dass die Kante auch in G^m enthalten ist. Für die Kante (v_i, v_j) gilt nach Konstruktion von G^* in Theorem 3.1.7, dass $(v_i, v_j) \in \overrightarrow{E_{G^*}}$. Falls $(v_i, v_j) \in \overrightarrow{E_G}$, so gilt nach Definition $\{v_i, v_j\} \in E_{G^m}$. Gilt $(v_i, v_j) \notin \overrightarrow{E_G}$, so wurde die Kante (v_i, v_j) im dritten Algorithmusschritt entfernt. Eine Kante (v_i, v_j) wird

aber nur dann entfernt, wenn v_i und v_j ein gemeinsames Kind v_k in G^* haben mit $k > j > i$, also $(v_i, v_k) \in \overrightarrow{E_{G^*}}$ und $(v_j, v_k) \in \overrightarrow{E_{G^*}}$. Da nach Konstruktion von G aber beim Eleminieren der Kanten bei dem Knoten mit dem größten Label gestartet wird und dann die Reduktion iterativ bei den Knoten mit dem nächst kleineren Label fortgeführt wird, sind die Kanten (v_i, v_k) und (v_j, v_k) auch in G vorhanden. Aus diesem Grund entsteht die Kante $\{v_i, v_j\}$ durch Moralisierung und es gilt $\{v_i, v_j\} \in G^m$.

q.e.d.

Schritt 3 und 4 im vorgestellten Algorithmus sind nicht notwendig, um eine Prämoralisierung zu erhalten. In Satz 4.2.5 wird aber gezeigt, dass es von Vorteil sein kann, wenn der erzeugte prämoralisierte Graph G möglichst wenig Kanten enthält.

3.2.2 Algorithmus für beliebige prämoralisierbare Graphen

Wie gezeigt ist die Menge der zerlegbaren Graphen echt enthalten in der Menge der prämoralisierbaren Graphen. Somit ist der beschriebene Algorithmus I nicht in der Lage, für alle prämoralisierbaren Graphen eine Prämoralisierung zu finden.
Im folgenden Abschnitt wird ein Algorithmus beschrieben, der sich auch für nicht zerlegbare Graphen einsetzen lässt. Dieser Algorithmus hat den Nachteil, dass er im schlechtesten Fall sehr viel mehr Rechenzeit benötigt als der gerade vorgestellte Algorithmus I für zerlegbare Graphen. Aus diesem Grund bietet es sich an, für zerlegbare Graphen Algorithmus I zu verwenden und für nicht zerlegbare Graphen den Algorithmus II des nächsten Abschnitts.

Algorithmus II: Prämoralisierungsalgorithmus für beliebige Graphen

Gegeben ist ein beliebiger ungerichteter Graph $H = (V_H, E_H)$. Definiere den gerichteten Graphen $G = (V_H, \emptyset)$ und die *Möglichkeitenmenge* $M = \emptyset \subseteq E_H$. In jedem Algorithmusschritt wird H um einen Knoten und/oder einige Kanten reduziert, während zusätzlich gerichtete Kanten in den Graphen G eingefügt werden können und auch die Menge M um Kanten aus H vergrößert werden kann. Abbildung 3.5 zeigt ein Schaubild des Algorithmus, der nun detailliert beschrieben wird. Im Folgenden bezeichnet (H_i, M_i, G_i) das Tripel vor einem Algorithmusschritt und $(H_{i+1}, M_{i+1}, G_{i+1})$ das Tripel nach einem Algorithmusschritt. Für alle i gilt, dass $V_{H_i} = V_{G_i}$ und $M_i \subseteq E_{H_i}$. Gestartet wird wie oben beschrieben mit

$$(H_0, M_0, G_0) = (H, \emptyset, G).$$

1. Falls es in H_i keinen Knoten gibt, so ist der Algorithmus erfolgreich gewesen. Der gesuchte Graph ist der Graph G_i.

2. Falls es in M_i Kanten $\{v, w\}$ gibt, bei denen v und w entweder keinen gemeinsamen Nachbarn in H_i haben oder alle gemeinsamen Nachbarn z von v und w in H_i auch gemeinsame Nachbarn von v und w in dem Teilgraphen von H_i sind, welcher nur aus Kanten von M_i besteht, so werden diese Kanten aus M_i und H_i gelöscht. Sei

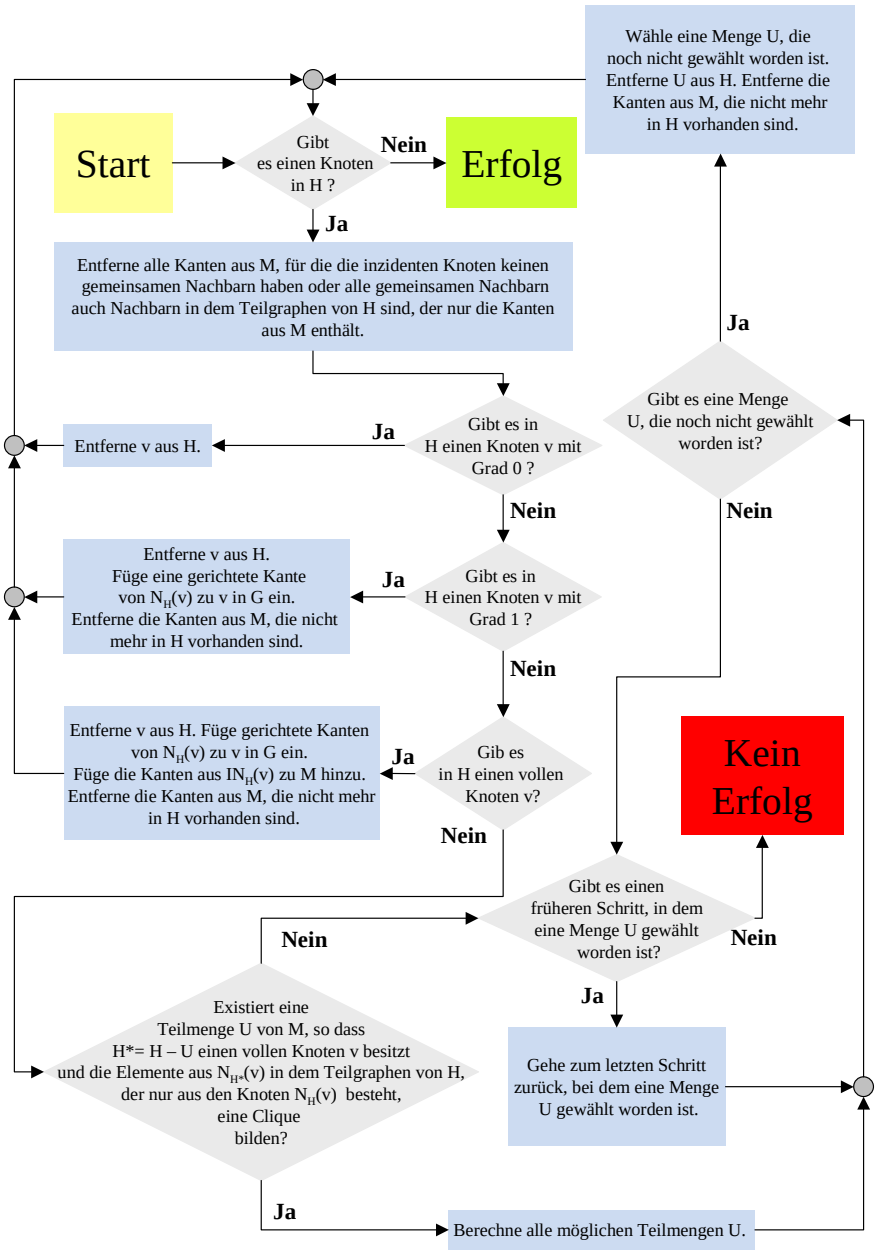

Abbildung 3.5: Die Abbildung verdeutlicht die einzelnen Schritte des Algorithmus II.

also $\hat{H}_i = (V_{H_i}, M_i)$ und $W := \{\{v, w\} \in M_i \mid N_{H_i}(v) \cap N_{H_i}(w) = \emptyset \vee$ für alle $z \in N_{H_i}(v) \cap N_{H_i}(w)$ gilt $z \in N_{\hat{H}_i}(v) \cap N_{\hat{H}_i}(w)\}$.

- $H_{i+1} = H_i - W$
- $M_{i+1} = M_i \backslash W$
- $G_{i+1} = G_i$

3. Falls es in H_i einen Knoten v vom Grad 0 gibt, so wird dieser aus H_i entfernt. Es wird keine Kante in G_i hinzugefügt und die Menge M_i wird nicht verändert. Dann zurück zu Punkt 1.

   - $H_{i+1} = H_i - \{v\}$
   - $M_{i+1} = M_i$
   - $G_{i+1} = G_i$

4. Falls es in H_i einen Knoten v vom Grad 1 gibt, so wird dieser zusammen mit der Kante $\{v, w\}$ aus H_i entfernt, wobei $\{w\} = N_{H_i}(v)$. In G_i wird die gerichtete Kante (w, v) eingefügt. Dann zurück zu Punkt 1. Die Kante $\{v, w\}$ kann nicht in M_i liegen, denn dann wäre sie schon in einem obigen Schritt 2 entfernt worden, denn v und w haben keinen gemeinsamen Nachbarn.

   - $H_{i+1} = H_i - \{v\}$
   - $M_{i+1} = M_i$
   - $G_{i+1} = G_i \cup \{(w, v)\}$

5. Falls es in H_i einen vollen Knoten v mit $\deg_{H_i}(v) \geq 2$ gibt, wird v aus H_i entfernt. Die Kanten, die sich zwischen den Nachbarknoten von v befinden, werden in die Möglichkeitenmenge M_i aufgenommen. Im gerichteten Graphen G_i werden gerichtete Kanten (x, v) mit $x \in N_{H_i}(v)$ eingefügt. Dann zurück zu Punkt 1.

   - $H_{i+1} = H_i - \{v\}$
   - $M_{i+1} = (M_i \cup IN_{H_i}(v)) \cap E_{H_{i+1}}$
   - $G_{i+1} = G_i \cup \{(x, v) \mid x \in N_{H_i}(v)\}$

6. Falls $M_i \neq \emptyset$ wird für jeden Knoten v, der Endpunkt mindestens einer Kante aus M_i ist, folgendes überprüft: Sei $N := H_i - (V_{H_i} \backslash N_{H_i}(v))$ (Teilgraph von H_i, der nur Knoten aus $N_{H_i}(v)$ enthält). Ist es möglich, eine Menge $W \subseteq M_i$ aus H_i zu entfernen, so dass der Knoten v in dem resultierenden Graphen $H_{i+1} = H_i - W$ nur noch mit solchen Knoten verbunden ist, die in N eine Clique bilden? Insbesondere ist der Knoten v dann voll in H_{i+1}. Alle möglichen Mengen $W \subseteq M_i$, für die dies zutrifft, werden notiert. Wenn es eine oder mehrere Mengen W gibt, so wird eine der möglichen Mengen W gewählt und weiter mit Punkt 8.

7. Falls keine Menge W gewählt werden kann, wird der letzte Index i betrachtet, an dem man die Auswahl zwischen mehreren möglichen Mengen $W \subseteq M_i$ hatte und noch nicht alle Möglichkeiten getestet worden sind, falls ein solcher Index existiert. In dem Fall wird das Tripel (H_i, M_i, G_i) in den Zustand zurück gesetzt, in dem es sich befand, bevor an dieser Stelle eine Menge W entfernt wurde. Es wird eine Menge W gewählt, die man noch nicht getestet hat, und weiter mit Punkt 8. Falls so eine Stelle nicht existiert, folgt Punkt 9.

8. Falls eine Menge W gewählt wurde, wird diese aus H_i und M_i entfernt und zurück zu Punkt 1.

   - $H_{i+1} = H_i - W$
   - $M_{i+1} = M_i \cap E_{H_{i+1}}$
   - $G_{i+1} = G_i$

9. Falls der Algorithmus diesen Schritt erreicht, bricht er ab. Der ursprüngliche Graph H ist nicht prämoralisierbar.

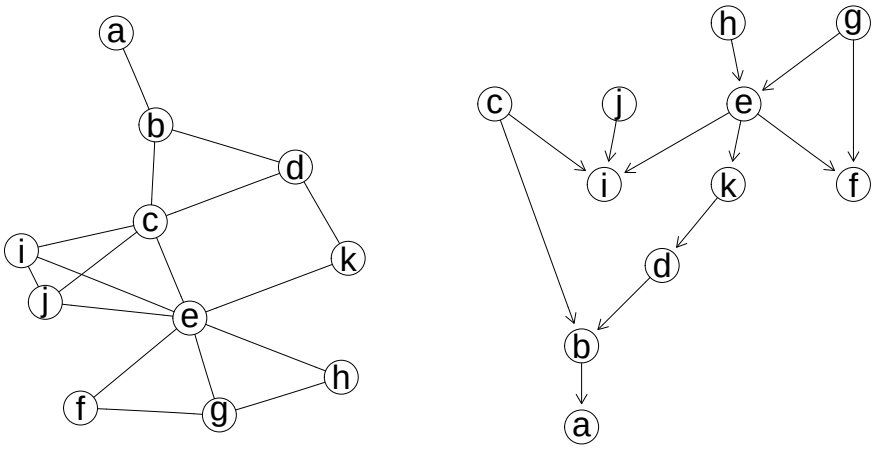

Abbildung 3.6: Ein nicht zerlegbarer aber prämoralisierbarer Graph und eine zugehörige Prämoralisierung durch Algorithmus II.

Die Funktionsweise des Algorithmus wird nun an einem Beispiel verdeutlicht. Abbildung 3.6 zeigt einen nicht zerlegbaren Graphen und eine zugehörige Prämoralisierung, die vom Algorithmus II erzeugt worden ist. Abbildung 3.7 zeigt für das gleiche Beispiel die einzelnen Reduktionen des Originalgraphen H. Abbildung 3.8 zeigt die einzelnen Schritte, in denen der Graph G erweitert wird.

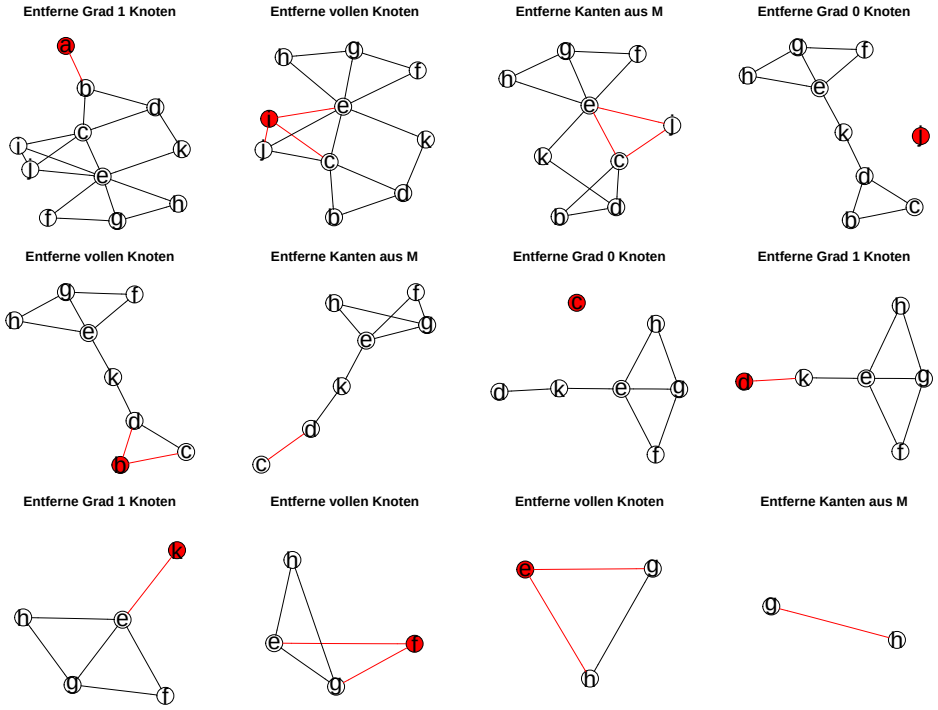

Abbildung 3.7: Beispiel eines prämoralisierbaren Graphen H und die Entstehung einer Prämoralisierung G durch Algorithmus II. Gezeigt ist die Reduktion des Graphen H. Rot kennzeichnet die Elemente, die im aktuellen Schritt eleminiert werden.

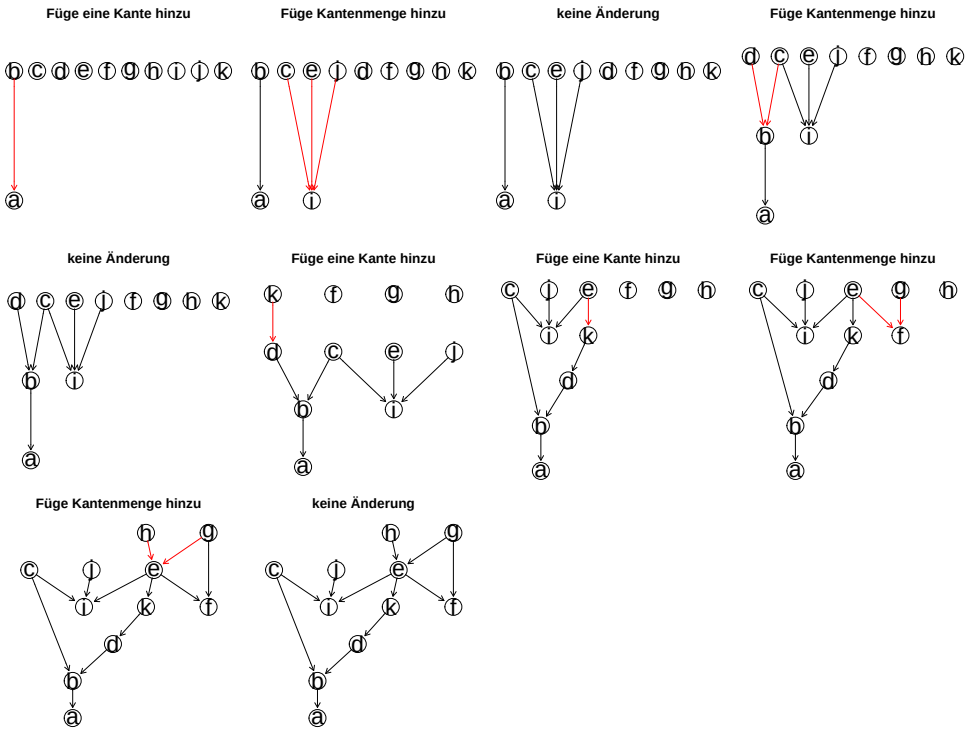

Abbildung 3.8: Beispiel eines prämoralisierbaren Graphen H und die Entstehung einer Prämoralisierung G durch Algorithmus II. Gezeigt ist der Aufbau des gerichteten akyklischen Graphen G. Rote Kanten werden in dem jeweiligen Schritt eingefügt.

Die folgenden Beispiele sollen einzelne Schritte des Algorithmus motivieren und erklären. Zuerst erfolgt die Motivation für die Möglichkeitenmenge M. Diese Menge wird eingeführt, weil man bei der Reduktion eines vollen Knotens v aus H_i mit $\deg_{H_i}(v) \geq 2$ nicht direkt festlegen kann, ob Kanten zwischen den Nachbarn von v, also Elemente aus $IN_{H_i}(v)$, aus H_i entfernt werden sollen oder nicht. Alle Elemente aus $IN_{H_i}(v)$ entstehen per Definition bei dem Moralisierungsvorgang von G_i. Sie müssen also nicht notwendigerweise durch gerichtete Kanten in der finalen Prämoralisierung G repräsentiert sein. Eine naheliegende Vorgehensweise wäre also, diese Kanten zu entfernen. Das Beispiel in Abbildung 3.9 zeigt, warum dies nicht immer möglich ist. Hier muss in Beispiel A die Kante $\{c, d\}$, die sich zwischen Nachbarn des zu entfernenden vollen Knotens e befindet, entfernt werden, damit der Algorithmus schließlich terminiert. In Beispiel B darf die Kante $\{c, e\}$ allerdings nicht eleminiert werden, damit der Algorithmus terminiert, weil man die Kante $\{c, e\}$ noch benötigt, um mit ihrer Hilfe die Kante $\{c, d\}$ in einem späteren Schritt zu eleminieren.

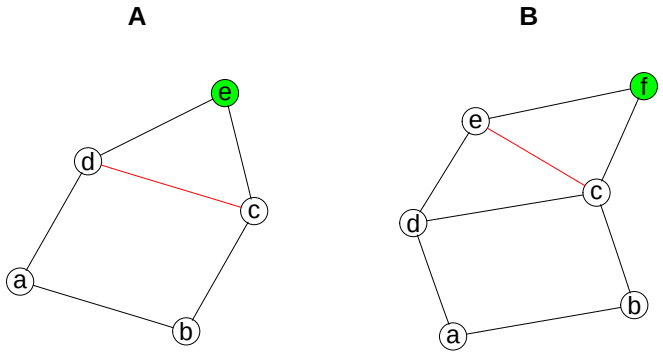

Abbildung 3.9: Zwei prämoralisierbare Graphen, bei denen je ein voller Knoten, gekennzeichnet durch die grüne Farbe, existiert, der in dem nächsten Algorithmusschritt eleminiert wird. Die roten Kanten können in den darauffolgenden Schritten entfernt werden.

Allgemein tritt also sowohl der Fall auf, dass eine Kante, die sich zwischen Nachbarn eines vollen Knotens befindet, auch in der finalen Prämoralisierung vorhanden ist, diese also nicht in einem Schritt i gelöscht werden darf, als auch der Fall, dass die Kante gelöscht werden muss. Aus diesem Grund wird die Möglichkeitenmenge eingeführt, die zuerst alle Kanten speichert, die später bei Bedarf gelöscht werden können. Dieser Bedarf tritt ein, wenn sich der Algorithmus an einer Stelle i befindet, wo es keinen vollen Knoten mehr in H_i gibt.

Ist man bei der Suche nach einer Prämoralisierung von H an einem Schritt angelangt, in dem der Graph H_i keinen vollen Knoten mehr hat, müssen Elemente aus M_i entfernt werden, bis wieder ein voller Knoten entsteht. Die Schritte 6-8 sorgen dafür, dass alle möglichen notwendigen Reduktionen getestet werden. Nun liegt die Frage nahe, ob dieses Vorgehen nötig ist, oder ob es ausreicht, eine beliebige Menge zu entfernen, so dass nach Entfernung der Menge ein voller Knoten entsteht. Das Beispiel in Abbildung 3.10 zeigt,

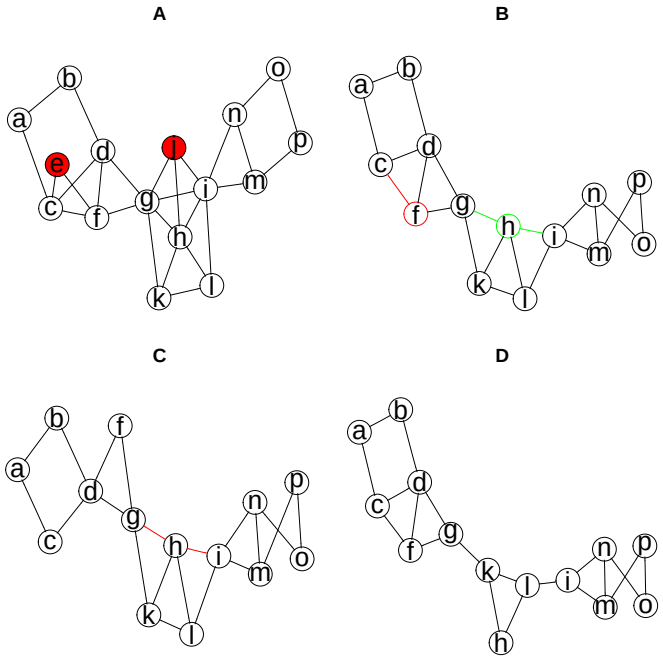

Abbildung 3.10: Beispiel für einen prämoralisierbaren Graphen(A) und zwei mögliche Reduktionen(C und D) der Möglichkeitenmenge M.

dass man keine beliebige Menge entfernen kann. In diesem Beispiel ist ein Graph gegeben, bei dem man verschiedene Möglichkeiten hat, Elemente aus M zu entfernen. Bild A zeigt einen Graphen mit den vollen Knoten e und j. Diese werden entfernt zusammen mit den Kanten, die auf jeden Fall entfernt werden dürfen (siehe Schritt 2). Es entsteht der Graph aus Abbildung B. Bei diesem Graphen hat man die Wahl, entweder die Kante $\{c, f\}$ oder die Kanten $\{h, g\}$ und $\{h, i\}$ zu entfernen. Abbildung C zeigt den Graphen nach Entfernung der Kante $\{c, f\}$. Der Graph ist nicht prämoralisierbar, selbst wenn man noch weitere der möglichen Kanten aus M entfernt, denn er enthält einen Zykel, bei dem keine im Zykel benachbarten Knoten mit gemeinsamem Nachbarn existieren (siehe Satz 3.1.6). Abbildung D zeigt den Graphen nach Wegnahme der Kanten $\{h, g\}$ und $\{h, i\}$. Dieser Graph ist prämoralisierbar.

Das Beispiel zeigt vor allem, dass die Kantenmenge, die man entfernt, nicht beliebig ist. Wenn man also durch die Entfernung einer Kantenmenge nicht zu einer Prämoralisierung kommt, so kann dies durch Entfernung einer anderen Menge der Fall sein. Das Beispiel zeigt weiterhin, dass eine Kantenmenge, die man entfernen muss, um zu einer Prämoralisierung zu kommen, nicht notwendigerweise weniger Elemente enthält hat als die anderen möglichen Kantenmengen.

Das Testen der möglichen Mengen (Schritt 6-8) ist der rechenintensivste Schritt im Algorithmus. Man ist deshalb daran interessiert, die Anzahl der zu testenden Möglichkeiten einzuschränken. Deshalb werden in Schritt 6 weitere Bedingungen an die zu entfernende

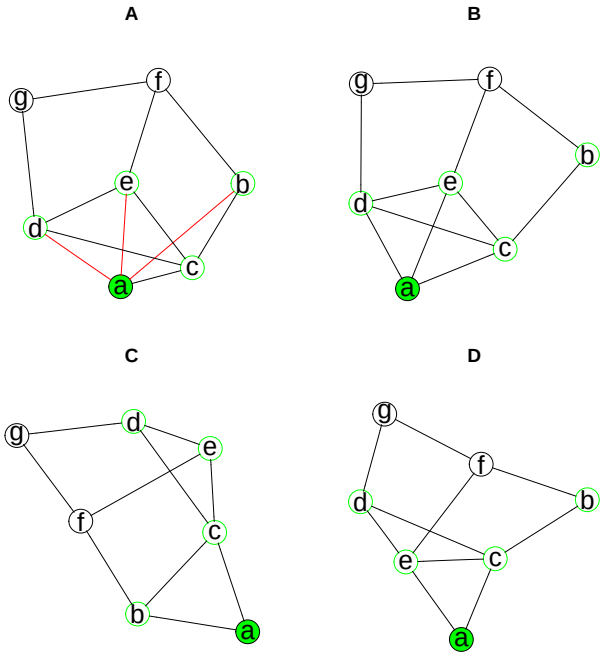

Abbildung 3.11: Beispiel eines Graphen H(A), bei dem es keinen vollen Knoten mehr gibt und man somit in Schritt 6 Kanten aus M entfernen möchte. Die Kanten in M sind rot gekennzeichnet. Die Nachbarschaft von a in H ist durch die grüne Umrandung gekennzeichnet.

Menge W gestellt. Diese werden in Abbildung 3.11 verdeutlicht, die ein Beispiel zeigt, bei dem alle möglichen in Schritt 6 zu untersuchenden Reduktionen der Menge M betrachtet werden. In Abbildung A ist ein Graph H gegeben, bei dem es keinen vollen Knoten mehr gibt und man somit in Schritt 6 Kanten aus M entfernen möchte. Die Kanten in M sind rot gekennzeichnet. Alle Kanten aus M sind zu dem Knoten a inzident, man betrachtet also a zusammen mit der Nachbarschaft aus a in H. Diese ist durch eine grüne Umrandung gekennzeichnet. In der Nachbarschaft von a in H gibt es zwei Cliquen, zum einen $\{c, d, e\}$ und zum anderen $\{b, c\}$. Man entfernt nun nur solche Kanten aus H, so dass a nur noch mit einer dieser beiden Cliquen verbunden ist. In Abbildung B wird die Kante $\{a, b\}$ entfernt und in Abbildung C die Kanten $\{a, e\}$ und $\{a, d\}$. Dies sind die beiden einzigen Fälle, die man betrachten muss. Es ist somit beispielsweise nicht nötig, den Fall zu betrachten, in dem die Kanten $\{a, b\}$ und $\{a, d\}$ entfernt werden (Abbildung D). Hier ist zwar a nach Entfernung der Kanten auch ein voller Knoten, aber die (neuen) Nachbarn von a bilden keine Clique in der ursprünglichen Nachbarschaft von a.

Die Anzahl der zu testenden Mengen in Schritt 6 wird auch reduziert, falls man die Menge M direkt reduzieren kann. Eine Kante lässt sich aus der Menge M entfernen, falls diese nicht mehr für eine Moralisierung genutzt werden kann. Kriterien hierfür werden in Schritt 2 des Algorithmus formuliert und angewandt. Beispielsweise kann eine Kante

$\{v, w\} \in M$ entfernt werden, falls v und w keinen gemeinsamen Nachbarn besitzen. In einem solchen Fall ist es nicht nötig, die Kanten (v, w) oder (w, v) in den aufzubauenden gerichteten Graphen G einzufügen, denn weder durch (v, w) noch durch (w, v) wird eine zusätzliche Kante durch Moralisierung entstehen. Abbildung 3.12 veranschaulicht die Kriterien, die in Algorithmusschritt 2 benutzt werden. In Beispiel A kann man beispielsweise die Kante $\{a, c\}$ entfernen, da alle gemeinsamen Nachbarn der Kante (e und f) nur durch Kanten aus M mit $\{a, c\}$ verbunden sind. Aus einem ähnlichen Grund können auch die Kanten $\{c, e\}, \{a, e\}, \{c, f\}$ und $\{a, f\}$ entfernt werden. In Beispiel B kann man die Kante $\{c, e\}$ nicht entfernen, da hier a ein gemeinsamer Nachbar dieser Kante ist, die Kante $\{a, e\}$ aber nicht in M liegt. Auch die Kante $\{a, c\}$ darf nicht entfernt werden.

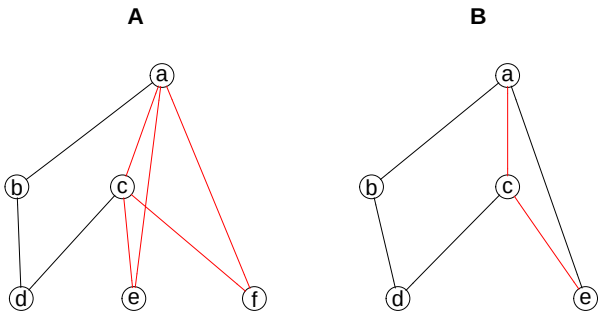

Abbildung 3.12: Beispiel für einen Graphen, bei dem Kanten aus der Möglichkeitenmenge entfernt werden dürfen, und einen Graphen, bei dem das nicht möglich ist. Bei diesen Graphen sind die Kanten aus M rot eingezeichnet.

Der übrige Teil dieses Kapitels beschäftigt sich mit dem Beweis, dass der vorgestellte Algorithmus auch wirklich eine Prämoralisierung findet, falls eine solche existiert. Das heißt zum Einen, dass der Algorithmus nicht frühzeitig abbrechen darf, das bedeutet in Schritt 9 gelangt und beendet wird. Zum Anderen soll der erzeugte Graph G wirklich eine Prämoralisierung des Graphen H sein. Für den Beweis müssen die folgenden vorbereitenden Lemmata gezeigt werden.

Lemma 3.2.2 *Sei $H = (V_H, E_H)$ ein prämoralisierbarer Graph mit $V_H \neq \emptyset$. Sei $v \in V_H$ ein beliebiger voller Knoten in H. Dann gibt es eine Prämoralisierung G von H, bei der v keine Kinder hat (das heißt $C_G(v) = \emptyset$), und für die gilt: $\{w, v\} \in E_H \Longrightarrow (w, v) \in \overrightarrow{E_G}$.*

Beweis: Sei v ein beliebiger voller Knoten in H. Ein solcher Knoten muss nach Satz 3.1.5 existieren, da H prämoralisierbar ist. Sei G' eine Prämoralisierung von H. Man betrachtet nun den Knoten v in dieser Prämoralisierung. Ist dieser Knoten nicht Elternteil eines weiteren Knotens, das heißt $C_G(v) = \emptyset$, und gilt $\{w, v\} \in E_H \Longrightarrow (w, v) \in \overrightarrow{E'_G}$, so ist man fertig. Das ist insbesondere der Fall, wenn $N_{G'}(v) = N_H(v) = \emptyset$. Sind die Bedingungen nicht erfüllt, so muss man die Prämoralisierung ändern. Hierzu betrachtet

man die Menge $U := v^+$ der von v ausgehenden Kanten in G. Die Kanten dieser Menge werden aus G' entfernt und durch die inversen Kanten ersetzt, also $G^* = G' - U \cup U'$ mit $U' = \{(w, v) \mid (v, w) \in U\}$. Nun werden noch die Kanten aus $I_H(v)$ eingefügt, die sich noch nicht in G^* befinden, also $G := G^* \cup \{(w, v) \mid \{w, v\} \in I_H(v)\}$. Der so entstandene Graph G ist wieder ein DAG. Man muss jetzt noch zeigen, dass G auch eine Prämoralisierung von H ist.

G unterscheidet sich von der Prämoralisierung G' nur durch die Richtung der Kanten von v zu anderen Knoten w und eventuell einiger eingefügter Kanten, die in H vorhanden sind und die auf v zeigen. Falls G keine Prämoralisierung ist, so liegt der Grund dafür entweder darin, dass durch das Drehen der Kanten oder das Hinzufügen der Kanten eine neue Kante durch Moralisierung entsteht, die nicht in H enthalten ist, oder dass durch das Drehen einer Kante eine Kante wegfällt, die aber in H enthalten ist und durch Moralisierung entstanden ist. Man betrachtet folgende Fälle:

- Für den Knoten v gilt $\deg_H(v) = 1$. Sei $\{v, w\} \in E_H$ die einzige zu v inzidente Kante in H. Diese Kante kann nicht durch Moralisierung entstehen, denn dazu muss jeder Knoten, der mit dieser Kante inzident ist, mindestens vom Grad 2 sein. Deshalb muss die Kante schon in G' vorhanden sein, also muss gelten $(v, w) \in \overrightarrow{E_{G'}}$. Diese Kante wird gedreht und es entsteht G. Durch Invertieren der einzigen Kante kann keine neue Kante durch Moralisierung entstehen, weil dazu v Kind einer weiteren Kante sein müsste. Es kann aber auch keine vorhandene Kante wegfallen, denn das wäre eine durch Moralisierung entstandene Kante, bei der v ein Endknoten ist, aber v ist nur vom Grad 1, und die einzige Kante $\{v, w\}$ ist nicht durch Moralisierung entstanden.

- Für den Knoten v gilt $\deg_H(v) \geq 2$. Da der Knoten v voll bezüglich H ist, bilden die Nachbarknoten $N_H(v)$ eine vollständige Menge in H. Für die durch Einfügen oder Invertieren der Kanten zusätzlich durch Moralisierung von G entstehende Kantenmenge W gilt aber $W \subseteq IN_H(v)$, und deshalb sind diese Kanten auch in H enthalten. Damit durch das Drehen einer Kante eine Kante wegfällt, die in H enthalten ist und durch Moralisierung entstanden ist, muss diese Kante v mit einem anderen Knoten w verbinden. All diese Kanten werden aber in die Prämoralisierung G eingefügt.

Damit ist gezeigt, dass auch G eine Prämoralisierung von H ist, und G erfüllt die geforderten Bedingungen.

q.e.d.

Lemma 3.2.3 *Sei $H = (V_H, E_H)$ ein prämoralisierbarer Graph mit $V_H \neq \emptyset$. Sei v ein beliebiger voller Knoten in H. Dann ist es möglich v aus H zu entfernen sowie eine Teilmenge der Kanten, die sich zwischen den Nachbarknoten von v befinden, so dass der dadurch entstehende Graph H' auch prämoralisierbar ist, das heißt es existiert eine Menge $W \subseteq IN_H(v)$, so dass $H' = (H - \{v\}) - W$ prämoralisierbar ist.*

Beweis: Nach Lemma 3.2.2 existiert eine Prämoralisierung G, bei der v nicht Elternteil eines anderen Knotens ist und für die gilt: $\{w, v\} \in E_H \Longrightarrow (w, v) \in \overrightarrow{E_G}$. Sei $G' = G - \{v\}$. G' ist immer noch ein DAG. Man betrachte die Moralisierung H' dieses neuen DAG. Diese Moralisierung unterscheidet sich zuerst von H durch das Fehlen des Knotens v und der zu v inzidenten Kanten. Zudem ist es auch möglich, dass sich die beiden Graphen durch eine Kantenmenge W unterscheiden, da durch Moralisierung von G Kanten entstanden sein können, die in G' nicht entstehen. Aber für alle sich zwischen G und G' unterscheidenden Kanten e gilt $e \in \{(w, v) | w \in N_H(v)\}$, also folgt $W \subseteq IN_H(v)$. Insgesamt ist H' also aus dem Graphen H entstanden durch Entfernen des Knotens v und einer Kantenmenge $W \subseteq IN_H(v)$.

q.e.d.

Es sei hier noch anzumerken, dass die in Satz 3.2.3 beschriebene Menge W auch die leere Menge sein kann. Dies ist insbesondere der Fall, falls der Knoten v keinen oder nur einen Nachbarn in H hat.

Lemma 3.2.4 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph mit $V_H \neq \emptyset$ und v ein voller Knoten in H. Sei eine Kantenmenge $U \subseteq E_H$ mit $H - U$ prämoralisierbar gegeben. Dann kann man eine Menge $U^* \subseteq U$ so wählen, dass v ein voller Knoten bezüglich $H^* := H - U^*$ ist und H^* ist prämoralisierbar.*

Beweis: Falls H prämoralisierbar ist, ist die Aussage trivial, da man in dem Fall $U^* = \emptyset$ wählen kann. Sei also H nicht prämoralisierbar, das heißt es gilt dann insbesondere $U \neq \emptyset$. Angenommen eine Menge U^* mit den obigen Eigenschaften existiert nicht, das heißt für jede Menge $U^* \subseteq U$, so dass $H^* = H - U^*$ prämoralisierbar ist, gilt, dass v kein voller Knoten bezüglich H^* ist. Insbesondere folgt daraus, dass $\deg_H(v) > 0$ sein muss.
Sei $\tilde{U} \subseteq U$ eine minimale Menge, so dass $H - \tilde{U}$ prämoralisierbar ist, das heißt für jede echte Teilmenge U' von $\tilde{U}$ gilt: $H - U'$ ist nicht prämoralisierbar. Auch für die minimale Menge $\tilde{U}$ gilt, dass v kein voller Knoten bezüglich $\tilde{H} := H - \tilde{U}$ ist, das heißt die Menge $\tilde{U}$ muss mindestens eine Kante $\{x, y\}$ enthalten, mit $x, y \in N_{\tilde{H}}(v)$ und $\{x, y\} \in IN_H(v)$. Diese Kante ist nicht mehr in $\tilde{H}$ vorhanden.
Sei $\tilde{G} = (V_{\tilde{G}}, \overrightarrow{E_{\tilde{G}}})$ eine Prämoralisierung von $\tilde{H}$. Diese wird so geändert, dass für alle $x \in N_{\tilde{H}}(v)$ eine Kante (x, v) eingefügt oder die Richtung einer existierenden Kante geändert wird. Das bedeutet $\bar{G} = (V_{\bar{G}}, \overrightarrow{E_{\bar{G}}})$ mit $V_{\bar{G}} = V_{\tilde{G}}$ und

$$\overrightarrow{E_{\bar{G}}} = \overrightarrow{E_{\tilde{G}}} - W_v \cup W_v^*$$

mit $W_v = \{(v, x) \,|\, x \in C_{\tilde{G}}(v)\}$ und $W_v^* = \{(x, v) \,|\, x \in N_{\tilde{H}}(v)\}$. Durch diese Modifikation werden insbesondere Kanten in $\tilde{G}$, die von v zu einem anderen Knoten w verlaufen, gedreht, so dass nun alle zu v inzidenten Kanten v als Endpunkt haben. Dadurch wird sichergestellt, dass der so erzeugte Graph wieder ein DAG ist. Sei $\bar{H}$ die Moralisierung von $\bar{G}$. v ist nach Konstruktion ein voller Knoten in $\bar{H}$. Man zeigt nun noch $\tilde{H} \subset \bar{H} \subseteq H$.

- $\tilde{H} \subset \bar{H}$: Angenommen es existiert eine Kante $e \in E_{\tilde{H}}$, mit $e \notin E_{\bar{H}}$. Da nach Konstruktion in $\bar{G}$ nur Kanten hinzugefügt oder gedreht werden, aber nicht entfernt, kann

dieser Fall nur eintreten, falls e durch Moralisierung in $\tilde{H}$ entstanden ist, dies aber bedingt durch das Drehen einer Kante in $\bar{H}$ nicht passiert. Aber Kanten, die auf solche Weise durch Moralisieren entstehen können, haben immer v als einen Endpunkt, und all diese Kanten sind schon in der Prämoralisierung $\bar{G}$ vorhanden und somit auch in $\bar{H}$. Die oben definierte Kante $\{x, y\}$ ist nicht in $\tilde{H}$ vorhanden, nach Konstruktion aber in $\bar{H}$, da dort alle Kanten aus $IN_{\tilde{H}}(v)$ vorhanden sind.

- $\bar{H} \subseteq H$: Angenommen es gibt eine Kante $e \in E_{\bar{H}}$, welche nicht in H vorhanden ist. Da $\tilde{H} \subseteq H$ gilt, kann es sich dabei nur um solch eine Kante handeln, die in $\bar{H}$ vorhanden ist, aber nicht in $\tilde{H}$. Das sind aber nach Konstruktion entweder Kanten $e \in I_{\tilde{H}}(v)$ oder Kanten $e \in IN_{\tilde{H}}(v)$. Aber all diese Kanten sind in H vorhanden, weil v ein voller Knoten in H ist und Nachbarn von v in $\tilde{H}$ auch Nachbarn von v in H sind.

Wegen $\tilde{H} \subset \bar{H} \subseteq H$ gibt es eine Menge $\bar{U} \subset \tilde{U}$ mit $H \backslash \bar{U} = \bar{H}$ und $\bar{H}$ ist prämoralisierbar. Das ist aber ein Widerspruch zur Minimalitätsannahme.

q.e.d.

Nach diesen Vorbereitungen kann mit dem Beweis begonnen werden. Der Beweis basiert hierbei auf folgender Idee: Man möchte für einen beliebigen Algorithmusschritt i zeigen, dass, wenn ein Graph H_i prämoralisierbar ist, so auch H_{i+1}. Das ist für den Schritt 5 des Algorithmus, die Reduktion um einen vollen Knoten v mit $\deg_{H_i}(v) > 1$, aber allgemein nicht gültig. Denn startet man mit einem prämoralisierbaren Graphen, so ist es möglich, dass nach der Reduktion des Graphen in einem Algorithmusschritt 5 dieser nicht mehr prämoralisierbar ist. Allerdings gibt es immer eine Menge $U \subseteq M$, so dass der Graph durch Entfernung der Menge U wieder prämoralisierbar wird (siehe Lemma 3.2.3). Aus diesem Grund ist folgende Definition für die weiteren Untersuchungen elementar.

Definition 3.2.5 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph und $M \subseteq E_H$. Dann heißt das Tupel (H, M) prämoralisierbar, falls eine Menge $U \subseteq M$ existiert mit $H - U$ ist prämoralisierbar. Sei Δ die Menge aller Tupel (H, M), die prämoralisierbar sind.*

Wenn man mit einem prämoralisierbaren Graphen H startet, so ist $(H, \emptyset) \in \Delta$. In den folgenden Lemmata wird gezeigt, dass die Algorithmusschritte 2-5 die Menge Δ nicht verlassen, das heißt, wenn ein Tupel vor dem Algorithmusschritt prämoralisierbar ist, so auch nach dem Algorithmusschritt. Zudem wird gezeigt, dass man bei Algorithmusschritt 6 eine Wahl der Reduktionsmenge W so treffen kann, dass man im darauffolgenden Algorithmusschritt 8 die Menge Δ auch nicht verlässt. Da man aber nicht weiß, welche Menge man nehmen muss, ist dies die Stelle, an der unter Umständen alle möglichen Mengen W getestet werden müssen.

Algorithmusschritt 2

Lemma 3.2.6 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph, $M \subseteq E_H$ und sei (H, M) prämoralisierbar. Sei W die Menge der Kanten in M, bei denen die inzidenten Knoten entweder keinen gemeinsamen Nachbarn haben, oder für jeden gemeinsamen Nachbarn der Endpunkte gilt, dass die Verbindungskanten von den Endpunkten zu dem gemeinsamen Nachbarn auch in M liegen. Sei also $\hat{H} = (V_H, M)$ und*

$$\begin{aligned} W := \quad & \{\{v,w\} \in M \mid N_H(v) \cap N_H(w) = \emptyset\} \\ \cup \quad & \{\{v,w\} \in M \mid \textit{für alle } z \in N_H(v) \cap N_H(w) \textit{ gilt } z \in N_{\hat{H}}(v) \cap N_{\hat{H}}(w)\}. \end{aligned}$$

Dann ist $(H - W, M \backslash W)$ prämoralisierbar.

Beweis: Es genügt, die Aussage für eine einelementige Menge $\{\{v,w\}\}$ zu zeigen. Denn falls W aus mehreren Kanten besteht, so kann man zuerst eine Kante $\{v_1, w_1\} \in W$ entfernen und erhält den Graphen $H_1 = H - \{\{v_1, w_1\}\}$. Für die übrigen Kanten aus W gilt aber auch in diesem Graphen, dass sie entweder keinen gemeinsamen Nachbarn haben oder aber alle Kanten zu diesen Nachbarn liegen in $M \backslash \{v,w\}$. Deshalb kann man dann die zweite Kante $\{v_2, w_2\} \in W$ aus H_1 entfernen. Diese Prozedur setzt man fort, bis keine Kante mehr im ursprünglichen W verblieben ist.

Für eine einelementige Menge $\{\{v,w\}\}$ muss die Existenz einer Menge $\tilde{U} \subseteq M \backslash \{\{v,w\}\}$ gezeigt werden mit $H - \{\{v,w\}\} - \tilde{U}$ ist prämoralisierbar. Da (H, M) prämoralisierbar ist, gibt es eine Menge $U \subseteq M$ mit $H^* := H - U$ ist prämoralisierbar. Man möchte nun zeigen, dass man eine Menge U immer so wählen kann, dass $\{v,w\} \in U$, denn dann ist die Aussage bewiesen, da man in dem Fall $\tilde{U} := U \backslash \{v,w\}$ wählen kann und somit wäre $H - \{v,w\} - \tilde{U} = H - U$ prämoralisierbar.

Der Beweis erfolgt durch Widerspruch: Man nimmt an, das $\{v,w\} \notin U$ für alle $U \subseteq M$ mit $H - U$ prämoralisierbar. Sei U nun maximal gewählt, das heißt $U \subseteq M$ mit $H - U$ ist prämoralisierbar und für alle $U \subset \bar{U} \subseteq M$ gilt, dass $H - \bar{U}$ nicht prämoralisierbar ist. Sei nun $H' := H - U$ für eine solche Menge U. Da auch für dieses maximale U gilt, dass $\{v,w\} \notin U$ folgt $\{v,w\} \in H'$. Es gilt:
Aussage 1: Man kann eine Prämoralisierung $G' = (V_{G'}, \overrightarrow{E_{G'}})$ von H' so wählen, dass $(v,w) \notin \overrightarrow{E_{G'}}$ und $(w,v) \notin \overrightarrow{E_{G'}}$.
Beweis der Aussage 1: Angenommen man hat eine Prämoralisierung G' mit $(v,w) \in \overrightarrow{E_{G'}}$ oder $(w,v) \in \overrightarrow{E_{G'}}$. Sei oBdA $(v,w) \in \overrightarrow{E_{G'}}$. Dann definiert man den DAG $G'' := G' - \{(v,w)\}$. Falls man zeigen kann, dass $(G'')^m = (G')^m$ ist man fertig, denn $(G')^m = H'$ und somit wäre G'' die gesuchte Prämoralisierung. Angenommen $(G')^m \neq (G'')^m$, so existiert eine nichtleere Menge von Kanten $K \subseteq E_{(G')^m} \backslash E_{(G'')^m}$ mit $E_{(G'')^m} \cup K = E_{(G')^m}$. Definiere $Z_{(v,w)} = \{\{z,v\} \in E_{H'} | (z,w) \in \overrightarrow{E_{G'}}\}$ als die Menge der Kanten aus H', die durch Moralisierung unter Mithilfe der Kante (v,w) entstehen könnten. Nach Voraussetzung gilt $Z_{(v,w)} \subseteq M$, denn für jede Kante $\{z,v\} \in Z_{(v,w)}$ gilt, dass z ein gemeinsamer

Nachbar von v und w ist, also nach Voraussetzung des Satzes $\{z, v\} \in M$. Somit gilt dann $K \subseteq \{\{v, w\}\} \cup Z_{(v,w)} \subseteq M$. Dann wäre aber $H' - K = H - (U \cup K)$ auch prämoralisierbar mit $U \cup K \subseteq M$ und $K \backslash U \neq \emptyset$. Das ist aber ein Widerspruch zur Maximalität von U und damit ist Aussage 1 bewiesen.

Sei $G' = (V_{G'}, \overrightarrow{E_{G'}})$ also eine Prämoralisierung von H' mit $(v, w) \notin \overrightarrow{E_{G'}}$ und $(w, v) \notin \overrightarrow{E_{G'}}$. Für diese Prämoralisierung betrachtet man die Menge

$$Z_{G'}^{\{v,w\}} = \{z \in V_{G'} | (v, z), (w, z) \in \overrightarrow{E_{G'}}\}.$$

Es gilt $Z_{G'}^{\{v,w\}} \neq \emptyset$, denn $(v, w) \notin \overrightarrow{E_{G'}}$ und $(w, v) \notin \overrightarrow{E_{G'}}$, aber $\{v, w\} \in E_{H'}$, denn es gilt $(G')^m = H'$. Also muss es mindestens einen Knoten $z \in V_{G'}$ geben mit $(v, z), (w, z) \in \overrightarrow{E_{G'}}$. Für alle $z \in Z_{G'}^{\{v,w\}}$ entfernt man nun die Kante (v, z). Sei also $F := \{(v, z) | z \in Z_{G'}^{\{v,w\}}\} \subseteq \overrightarrow{E_{G'}}$ und $G'' = G' - F$. G'' ist ein DAG. Für diesen DAG gilt

$$Z_{G''}^{\{v,w\}} = \{z \in V_{G''} | (v, z), (w, z) \in \overrightarrow{E_{G''}}\} = \emptyset$$

nach Konstruktion. Bezeichne $H'' = (G'')^m$. Es gilt $\{v, w\} \notin E_{H''}$, denn weder ist $(v, w) \in \overrightarrow{E_{G''}}$ noch $(w, v) \in \overrightarrow{E_{G''}}$, und die Kante kann auch nicht durch Moralisierung entstehen, denn $Z_{G''}^{\{v,w\}} = \emptyset$, also existiert kein Punkt $z \in V_{G''}$ mit $(v, z), (w, z) \in \overrightarrow{E_{G''}}$. Man setzt nun $R = E_{H'} \backslash E_{H''}$. Es gilt $R \neq \emptyset$ da $\{v, w\} \in R$. Zu zeigen bleibt nun noch:
Aussage 2: $R \subseteq M$.
Beweis von Aussage 2 Es gilt

$$R \subseteq \{\{v, w\}\} \cup \{\{v, z\} | z \in Z_{G'}^{\{v,w\}}\} \cup \{\{v, u\} | (v, z) \in F, (u, z) \in \overrightarrow{E_{G'}}\}$$

und $\{\{v, w\}\}$ und $\{\{v, z\} | z \in Z_{G'}^{\{v,w\}}\} \subseteq M$ nach Voraussetzung.
Um zu zeigen dass $\{\{v, u\} | (v, z) \in F, (u, z) \in \overrightarrow{E_{G'}}\} \subseteq M$ betrachtet man ein Element $\{v, u\}$ dieser Menge und einen zugehörigen Knoten z mit $(v, z) \in F$ und $(u, z) \in \overrightarrow{E_{G'}}$. Man möchte zeigen, dass u gemeinsamer Nachbar von v und w in H' und somit in H ist, denn dann ist die Aussage bewiesen, da dann nach Voraussetzung des Satzes gilt, dass $\{v, u\} \in M$. Da $\{v, u\} \in E_{H'}$ reicht es also zu zeigen, dass $\{w, u\} \in E_{H'}$. Da aber $(v, z) \in F$ existiert auch die Kante $(w, z) \in \overrightarrow{E_{G'}}$. Falls die Kanten (w, u) und (u, w) also nicht im Graphen G' wären, so würde die Kante $\{u, w\}$ auf jeden Fall durch Moralisierung entstehen. Somit ist die Aussage 2 bewiesen.

Also hat man gezeigt, dass $R \subseteq M$ und für den prämoralisierbaren Graphen H'' gilt $H'' = H' - R = H - (U \cup R)$ mit $U \cup R \subseteq M$ und $R \backslash U \neq \emptyset$, da $\{v, w\}$ in R aber nicht in U. Das ist aber ein Widerspruch zur Maximalität von U.

q.e.d.

Algorithmusschritt 3

Lemma 3.2.7 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph, $M \subseteq E_H$ und sei (H, M) prämoralisierbar. Sei $v \in V_H$ ein Knoten mit $\deg_H(v) = 0$. Dann ist $(H - \{v\}, M)$ prämoralisierbar.*

Beweis: Da (H, M) prämoralisierbar ist, existiert eine Menge $U \subseteq M$ mit $H^* = H - U$ ist prämoralisierbar. Sei G^* eine Prämoralisierung von H^*. Es ist $\deg_{G^*}(v) = \deg_{H^*}(v) = \deg_H(v) = 0$. Definiere $\tilde{G} = G^* - \{v\}$. Dann ist $(\tilde{G})^m = H^* - \{v\} = H - \{v\} - U$ und die Aussage ist bewiesen.

q.e.d.

Algorithmusschritt 4

Lemma 3.2.8 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph, $M \subseteq E_H$ und sei (H, M) prämoralisierbar. Sei $v \in V_H$ ein Knoten mit $\deg_H(v) = 1$. Definiere $H^* = H - \{v\}$. Dann ist $(H^*, M \cap E_{H^*})$ prämoralisierbar.*

Beweis: Da (H, M) prämoralisierbar ist, existiert eine Menge $U \subseteq M$ mit $H^* = H - U$ ist prämoralisierbar. Da $\deg_H(v) = 1$ ist v ein voller Knoten. Nach Lemma 3.2.4 kann man U so wählen, dass v auch noch ein voller Knoten in H^* ist, das bedeutet in diesem Fall $\deg_{H^*}(v) = 1$. Also existiert ein $w \in V_H$ mit $\{v, w\} \in E_{H^*}$. Sei G^* eine Prämoralisierung von H^*. Diese kann man nach Lemma 3.2.2 so wählen, dass $(w, v) \in \overrightarrow{E_{G^*}}$ und es gilt $\deg_{G^*}(v) = 1$, das heißt es gibt keine weitere Kante, die von v ausgeht oder zu v läuft. Definiere $G := G^* - \{v\}$. Dann ist $G^m = H^* - \{v\} = H - \{v\} - U$.

q.e.d.

Algorithmusschritt 5

Lemma 3.2.9 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph, $M \subseteq E_H$ und (H, M) sei prämoralisierbar. Sei $v \in V_H$ ein voller Knoten in H und $\deg_H(v) \geq 2$. Definiere $H^* := H - v$ und $M^* := M \backslash I_H(v) \cup IN_H(v)$. Dann ist (H^*, M^*) prämoralisierbar.*

Beweis: Falls H prämoralisierbar ist, folgt die Aussage aus Lemma 3.2.3. Sei also H nicht prämoralisierbar. Da $IN_H(v) \cap I_H(v) = \emptyset$ und $(M \backslash I_H(v)) \cap I_H(v) = \emptyset$ gilt $M^* \subseteq E_H \backslash I_H(v) \subseteq E_{H^*}$.
Man muss noch zeigen, dass eine Menge $U^* \subseteq M^*$ existiert mit $H^* - U^*$ prämoralisierbar. Da (H, M) prämoralisierbar ist, existiert eine Menge $U \subseteq M$ mit $\tilde{H} = H - U$ prämoralisierbar. Da H nicht prämoralisierbar ist, gilt insbesondere $U \neq \emptyset$. Nach Lemma 3.2.4 kann man die Menge U so wählen, dass v ein voller Knoten bezüglich $\tilde{H}$ ist. Man kann also nach Lemma 3.2.3 den Knoten v aus $\tilde{H}$ entfernen zusammen mit einer Menge $N \subseteq IN_{\tilde{H}}(v)$, und der so entstehende Graph $\hat{H} = (V_{H^*}, (E_{\tilde{H}} \backslash I_{\tilde{H}}(v)) \backslash N)$ ist prämoralisierbar.
Definiere $U^* := (U \backslash I_H(v)) \cup N$. Dann gilt $U^* \subseteq M^*$ weil $U \backslash I_H(v) \subseteq M \backslash I_H(v)$ und

$N \subseteq IN_{\tilde{H}}(v) \subseteq IN_H(v)$. Man möchte also noch zeigen, dass $H^* - U^*$ prämoralisierbar ist. Dazu reicht es zu zeigen, dass $H^* - U^* = \hat{H}$, denn $\hat{H}$ ist nach Konstruktion prämoralisierbar. Da $\hat{H}$ und H^* die gleichen Knoten besitzen, muss man nur die Kantenmengen betrachten. Es gilt:

$$\begin{aligned} E_{\hat{H}} &= (E_{\tilde{H}} \backslash I_{\tilde{H}}(v)) \backslash N \\ &= ((E_H \backslash U) \backslash I_{\tilde{H}}(v)) \backslash N \\ &= E_H \backslash (U \cup I_{\tilde{H}}(v) \cup N) \\ &= E_H \backslash (U \cup (I_H(v) \backslash U) \cup N) \\ &= E_H \backslash (I_H(v) \cup (U \backslash I_H(v)) \cup N) \\ &= (E_H \backslash I_H(v)) \backslash ((U \backslash I_H(v)) \cup N) \\ &= E_{H^*} \backslash ((U \backslash I_H(v)) \cup N) \\ &= E_{H^*} \backslash U^* \end{aligned}$$

Somit erfüllt U^* die beiden geforderten Bedingungen und (H^*, M^*) ist prämoralisierbar. *q.e.d.*

Algorithmusschritt 6

Zum 6. Algorithmusschritt gelangt man nur, wenn der Graph H keinen vollen Knoten mehr besitzt. Insbesondere ist dieser Graph nach Lemma 3.1.5 nicht mehr prämoralisierbar. Dann muss man aus diesem Graphen eine Teilmenge $A \subseteq M$ entfernen, die nun genauer charakterisiert wird.

Lemma 3.2.10 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph, $M \subseteq E_H$ und (H, M) prämoralisierbar. In H gebe es keinen vollen Knoten. Es existiert eine Teilmenge $A \subseteq M$ mit folgenden Eigenschaften*

1. *$A \neq \emptyset$*
2. *Es existiert ein Knoten v in H mit $A \subseteq I_H(v)$ und dieser Knoten ist voll bezüglich $H^* := H - A$.*
3. *$(H^*, M \backslash A)$ ist prämoralisierbar.*
4. *$N_{H^*}(v)$ bilden eine Clique im Teilgraphen $H - (V_H \backslash N_H(v))$, der Teilgraph von H, der nur aus den Nachbarknoten von v besteht.*

Die Eigenschaften der Menge A implizieren zwei Dinge bei dem Algorithmusschritt 6. Man kann sich beim Testen der möglichen Teilmengen zum einen auf solche Mengen beschränken, bei der alle Kanten einen gemeinsamen Knoten v haben. Zudem muss man nur solche Teilmengen A betrachten, bei denen nach Wegnahme der Menge A der Knoten

v mit einer Menge von Knoten verbunden ist, die im Subgraphen der Nachbarknoten von v in H eine Clique bilden (siehe Abbildung 3.11).

Beweis: Da (H, M) prämoralisierbar ist, existiert eine Menge U mit $U \subseteq M$ und $\hat{H} = H - U$ ist prämoralisierbar. Weil aber H keinen vollen Knoten hat, ist H nach Satz 3.1.5 nicht prämoralisierbar, also gilt insbesondere $U \neq \emptyset$. Nach Satz 3.1.5 muss $\hat{H}$ einen vollen Knoten v enthalten. Man definiert $A := I_H(v) \cap U$. Man will für diese Menge A die Eigenschaften 1-3 zeigen.

Da v nicht voll in H gewesen ist, gilt $|I_H(v)| \geq 2$. Weil aber v ein voller Knoten in $\hat{H}$ ist, muss die Menge U Elemente aus $I_H(v)$ enthalten, denn wenn ein Knoten v nicht voll in einem Graphen H ist, aber voll ist, nachdem Kanten entfernt worden sind, so müssen unter den entfernten Kanten solche gewesen sein, die in H zu v inzident sind. Hieraus folgt insbesondere $A \neq \emptyset$, also ist Eigenschaft 1 gezeigt. Eigenschaft 2 folgt nach Konstruktion von A. Es gilt nun

$$U = A \cup (U \backslash I_H(v)).$$

Somit ist $\hat{H} = (H - A) - (U \backslash I_H(v))$. Da aber $(U \backslash I_H(v)) \subset M \backslash A$ gilt auch Eigenschaft 3.

Man wählt nun A minimal, das heißt für alle $B \subset A$ gilt, dass B nicht Eigenschaft 1,2 und 3 erfüllt. Man bezeichnet das minimale A mit $\tilde{A}$. Nun muss gezeigt werden, dass $\tilde{A}$ auch Eigenschaft 4 erfüllt, also nach Wegnahme der Menge $\tilde{A}$ der Knoten v mit solchen Knoten verbunden ist, die in dem Teilgraphen der Nachbarknoten von v in H eine Clique bilden.

Sei $\tilde{H} := H - \tilde{A}$. Angenommen die Nachbarknoten von v bilden keine Clique. Da die Menge aber vollständig ist - v ist ein voller Knoten in $\tilde{H}$ - muss die Maximalitätseigenschaft verletzt sein, das bedeutet, es muss mindestens einen Knoten $w \in N_H(v)$ geben, so dass $N_{\tilde{H}}(v) \cup \{w\}$ eine vollständige Menge in H bildet. Da $\{w, v\} \in E_H$ aber $\{w, v\} \notin E_{\tilde{H}}$ muss gelten $\{w, v\} \in \tilde{A}$. Man definiert $A^* = \tilde{A} \backslash \{\{w, v\}\}$. A^* ist nicht leer, denn in diesem Fall würde A nur aus dem Element $\{w, v\}$ bestehen. Wäre dies der Fall, müsste der Knoten v aber in H auch schon voll gewesen sein, da $N_{\tilde{H}}(v) \cup \{w\}$ eine vollständige Menge in H bildet. Das ist aber nach Voraussetzung nicht der Fall. Also erfüllt A^* Eigenschaft 1. A^* erfüllt auch Eigenschaft 2, da $N_{\tilde{H}}(v) \cup \{w\}$ eine vollständige Menge in H bildet und somit Knoten v voll ist bezüglich $H \backslash A^*$. Zu zeigen ist noch, dass $(H - A^*, M \backslash A^*)$ prämoralisierbar ist. Also muss ein $W \subseteq M \backslash A^*$ existieren mit $H - A^* - W$ prämoralisierbar.

Da $(\tilde{H}, M \backslash \tilde{A})$ prämoralisierbar ist, existiert eine Menge $\hat{U} \subseteq M \backslash \tilde{A}$ mit $\hat{H} = \tilde{H} - \hat{U}$ prämoralisierbar. Diese kann man nach Lemma 3.2.4 so wählen, dass v auch ein voller Knoten in $\hat{H}$ ist. Sei $\hat{G}$ eine Prämoralisierung von $\hat{H}$. Da v ein voller Knoten in $\hat{H}$ ist, kann man nach Lemma 3.2.2 eine Prämoralisierung so wählen, dass v kein Elternteil eines anderen Knotens ist. In $\hat{G}$ fügt man noch die gerichtete Kante (w, v) ein und erhält so den DAG $G^* = \hat{G} \cup \{(w, v)\}$. Man betrachtet nun die Moralisierung H^* von G^* und den prämoralisierbaren Graphen $\hat{H}$. Die Prämoralisierungen dieser beiden Graphen unterscheiden sich durch die Kante (w, v). Aus diesem Grund ist $\hat{H} \subset H^*$ mit $E_{H^*} \backslash E_{\hat{H}} = \{\{w, v\}\} \cup T$ und $T \subseteq IN_{H^*}(v)$. Man definiert $W := \hat{U} \backslash T$. Wegen $\hat{U} \subseteq M \backslash \tilde{A} \subseteq M \backslash A^*$ gilt $W \subseteq M \backslash A^*$. Zudem gilt

$$\begin{aligned} H^* &= \hat{H} \cup \{\{w,v\}\} \cup T \\ &= (\tilde{H} - \hat{U}) \cup \{\{w,v\}\} \cup T \\ &= (H - \tilde{A}) \cup \{\{w,v\}\} - \hat{U} \cup T \\ &= (H - (\tilde{A} \backslash \{\{w,v\}\})) - \hat{U} \cup T \\ &= (H - A^*) - \hat{U} \cup T \\ &= (H - A^*) - (\hat{U} \backslash T) \\ &= (H - A^*) - W \end{aligned}$$

weil $T \subseteq IN_{H^*}(v) \subseteq E_{H-A^*}$, $\{w,v\} \in \tilde{A}$ und deshalb insbesondere $\{w,v\} \notin \hat{U}$. Damit erfüllt also A^* die Eigenschaften 1,2 und 3 und es gilt $A^* \subset \tilde{A}$. Das ist aber ein Widerspruch zur Minimalität. Somit ist der Widerspruch zu der Annahme gezeigt, dass die Nachbarknoten von v keine Clique bilden.

q.e.d.

Das folgende Theorem fasst die in den letzten Sätzen gezeigten Sachverhalte zusammen und es kann somit gezeigt werden, dass Algorithmus II zu einem gegebenen prämoralisierbaren Graphen H eine Prämoralisierung findet.

Theorem 3.2.11 *Sei $H = (V_H, E_H)$ ein ungerichteter Graph. H ist prämoralisierbar genau dann wenn der Algorithmus II für H in Schritt 1 endet. Insbesondere ist der erzeugte gerichtete Graph $G = (V_G, \overrightarrow{E_G})$ eine Prämoralisierung von H.*

Beweis: Es wird zuerst gezeigt, dass der Algorithmus in Schritt 1 endet, falls H prämoralisierbar ist. In Algorithmus II wird der Graph H in jeder Iteration reduziert, entweder um einen Knoten oder um eine Menge von Kanten. Wenn der Algorithmus Kanten eleminiert, so wird sicher gestellt, dass in der nächsten Iteration wieder ein Knoten eleminiert wird. Also

$$|V_{H_{i+1}}| \leq |V_{H_i}|$$

und falls $|V_{H_{i+1}}| = |V_{H_i}|$ gilt

$$|V_{H_{i+2}}| < |V_{H_i}|.$$

In den Lemmata 3.2.6, 3.2.7, 3.2.8 und 3.2.9 ist gezeigt worden, dass der Algorithmus die Menge Δ in den Schritten 2-5 nicht verlässt. Für Schritt 8 ist in Lemma 3.2.10 gezeigt worden, dass immer eine Menge gefunden werden kann, bei der Δ nicht verlassen wird. Da aber gerade in jedem Schritt 6 immer alle möglichen Mengen getestet werden, betrachtet man nun immer den Fall, in dem in Schritt 8 die Menge entfernt wird, so dass auch in diesem Schritt die Menge Δ nicht verlassen wird. Es ergibt sich somit eine Folge

$$(H_0, M_0) \rightarrow (H_1, M_1) \rightarrow \ldots \rightarrow (H_n, M_n)$$

mit $(H_0, M_0) = (H, \emptyset)$, $H_n = (\emptyset, \emptyset)$ und $(H_i, M_i) \in \Delta$ für alle i. Das bedeutet, der Algorithmus reduziert den anfänglichen Graphen H auf einen Graphen ohne Knoten, falls der

ursprüngliche Graph prämoralisierbar ist, also endet der Algorithmus in Schritt 1. Insbesondere gibt es für jede Kante in H einen Algorithmusschritt, in dem die Kante eleminiert wird.

Es soll nun die Umkehrung des Satzes gezeigt werden. Hierzu sei H ein Graph, für den der Algorithmus in Schritt 1 endet. Man will zeigen, dass H prämoralisierbar ist. Dafür genügt es zu zeigen, dass die Moralisierung des durch den Algorithmus erzeugten gerichteten Graphen G gerade H ist. Nach Konstruktion von G ist dieser Graph ein DAG. Der gerichtete Graph G und somit auch die Moralisierung besitzt die gleichen Knoten wie der Graph H. Also bleibt zu zeigen $E_{G^m} = E_H$.

- $E_{G^m} \subseteq E_H$: Sei $\{v, w\} \in E_{G^m}$. Dann unterscheidet man zwei Fälle:

  1. $(v, w) \in \overrightarrow{E_G}$ oder $(w, v) \in \overrightarrow{E_G}$. Dann gilt aber nach Konstruktion $\{v, w\} \in E_H$, denn in G werden nur gerichtete Kanten eingefügt, wo sich die zugehörigen ungerichteten Kanten in H befindet.
  2. $\{v, w\}$ ist durch Moralisierung entstanden. Dann existiert aber ein $x \in V_G$ mit $(v, x), (w, x) \in \overrightarrow{E_G}$. Nach Konstruktion muss dann aber x in einem obigen Algorithmusschritt ein voller Knoten in einem Teilgraphen von H gewesen sein, das heißt es existiert ein Teilgraph $H^* \subseteq H$ mit $v, w \in N_{H^*}(x)$ und $N_{H^*}(x)$ ist eine vollständige Teilmenge. Dann gilt aber $\{v, w\} \in H^*$ und somit $\{v, w\} \in H$.

- $E_H \subseteq E_{G^m}$: Sei $\{v, w\} \in E_H$. Da der Algorithmus terminiert, gibt es einen Algorithmusschritt, in dem diese Kante aus H entfernt wird. Man unterscheidet zwei Fälle:

  1. Die Kante wird in Schritt 3,4, oder 5 entfernt. Nach Konstruktion wird dann aber die Kante (v, w) oder die Kante (w, v) in G eingefügt und somit gilt $\{v, w\} \in E_{G^m}$.
  2. Die Kante $\{v, w\}$ wird in einem Schritt 2 oder 8 entfernt. Damit die Kante in einem der beiden Schritte entfernt werden kann, muss sie sich in der Möglichkeitenmenge M befinden. Das bedeutet aber, dass es einen Algorithmusschritt gegeben hat, in dem ein voller Knoten u entfernt worden ist und w und v Nachbarknoten von u gewesen sind. Also existiert ein Teilgraph $H^* \subseteq H$ und ein Knoten $u \in V_G$ mit $v, w \in N_{H^*}(u)$ und $N_{H^*}(u)$ vollständige Teilmenge. In dem Schritt wurden die Kanten (v, u) und (w, u) in G eingefügt, also $(v, u), (w, u) \in G$. Hieraus folgt aber für die Moralisierung G^m dass $\{v, w\} \in E_{G^m}$.

q.e.d.

Mit Theorem 3.2.11 hat man nun die gewünschte notwendige und hinreichende Bedingung erhalten. Prämoralisierbare Graphen können dadurch charakterisiert werden, dass Algorithmus II in Schritt 1 beendet wird. Wie erwähnt ist der Algorithmus II allerdings im schlechtesten Fall sehr rechenintensiv, obwohl sich in der Praxis gezeigt hat, dass, falls

der Graph prämoralisierbar ist, der Algorithmus sehr schnell terminiert, da es augenscheinlich nur wenige Möglichkeiten in Schritt 6 gibt, die man nicht auswählen darf. Trotzdem müssen im schlechtesten Fall alle Moglichkeiten getestet werden, wie das beispielsweise in Abbildung 3.10 gezeigt worden ist.

Es sei hier noch angemerkt, dass man sich bei der Entscheidung, ob ein Graph prämoralisierbar ist oder nicht, auf die Untersuchungen der einzelnen Zusammenhangskomponenten beschränken kann, das heißt es gilt folgender Satz:

Satz 3.2.12 *Ein ungerichteter Graph H ist genau dann prämoralisierbar, falls jede Zusammenhangskomponente prämoralisierbar ist.*

Beweis: Die Aussage ist trivial. Falls jede Zusammenhangskomponente von H prämoralisierbar ist, so kann man die einzelnen Prämoralisierungen zu einer Prämoralisierung von H kombinieren. Ist $H = (V, E)$ prämoralisierbar, G eine Prämoralisierung von H und $\tilde{H} = (\tilde{V}, \tilde{E})$ eine Zusammenhangskomponente von H, so ist $\tilde{G} := G - (V \backslash \tilde{V})$ eine Prämoralisierung von $\tilde{H}$.

q.e.d.

Man kann also die Zusammenhangskomponenten getrennt voneinander betrachten und dann entweder Algorithmus I oder Algorithmus II anwenden, je nachdem, ob die Zusammenhangskomponente zerlegbar ist oder nicht. Dieses kann die Geschwindigkeit des Algorithmus stark verbessern.

Kapitel 4

Positiv definite Matrizen mit Nebenbedingungen

Dieses Kapitel beschäftigt sich mit der Erzeugung einer positiv definiten Matrix mit Nebenbedingungen und der daraus resultierenden Erzeugung von Werten einer multivariaten Normalverteilung. Positiv definite Matrizen mit Nebenbedingungen spielen eine zentrale Rolle in den Validierungsstrategien für Netzwerkalgorithmen. Für das in diesem Kapitel vorgestellte Verfahren zum Simulieren von positiv definiten Matrizen mit Nebenbedingungen ist die Konstruktion einer Prämoralisierung eines Graphen, falls diese möglich ist, der elementare Schritt, so dass der im letzten Kapitel vorgestellte Algorithmus hier eine Anwendung findet.

4.1 Definitionen und einige bekannte Ansätze

In diesem Abschnitt wird zuerst auf Schwierigkeiten hingewiesen, die bei der Erzeugung von positiv definiten Matrizen mit Nebenbedingungen auftreten können. Es folgen drei verschiedene Strategien, mit denen man positiv definite Matrizen mit Nebenbedingungen erzeugen kann.

Möchte man aus der Menge der positiv definiten Matrizen zufällige Elemente erzeugen, so existieren hierfür verschiedene Möglichkeiten. Zum Beispiel wurde in Satz 2.2.3 gezeigt, dass man ein Element aus der Menge der symmetrischen positiv definiten Matrizen erhalten kann, indem man eine zufällige obere Dreiecksmatrix $L = (l_{ij}) \in M(n, n, \mathbb{R})$ mit $l_{kk} > 0$ für alle k erzeugt. Man ist nun aber nicht daran interessiert, aus der gesamten Menge der symmetrischen positiv definiten Matrizen Daten zu generieren, sondern man möchte diese Menge einschränken. Es sollen einige vorher definierte Einträge der erzeugten Matrix Null sein. Diese Einschränkungen seien durch einen ungerichteten Graphen gegeben. Sei $H = (V, E)$ ein ungerichteter Graph mit Knotenmenge $V = \{1, \ldots, n\}$. Man definiert dann die Menge

$$SPR(n, H) := \{\Omega = (\omega_{ij}) \in SP(n) | \{i, j\} \notin E \Rightarrow \omega_{ij} = \omega_{ji} = 0 \text{ für alle } i, j\}.$$

$SPR(n, H)$ enthält demnach alle symmetrischen positiv definiten Matrizen, für die einige Einträge durch einen vorgegebenen Graphen auf 0 festgesetzt sind. Solche Matrizen zu erzeugen ist nicht trivial. Der Raum für die Einträge der Matrix, die keine Nebenbedingungen erfüllen müssen, ist auch einzuschränken, um die Definitheit zu gewährleisten. Speziell gilt es im Allgemeinen nicht, dass, wenn man mit einer symmetrischen positiv definiten Matrix startet und dann gewisse vorgeschriebene Elemente auf Null setzt, diese Matrix wieder positiv definit ist. Zur Verdeutlichung betrachtet man die Menge der symmetrischen Matrizen der Form

$$M = \begin{pmatrix} 1 & x_1 & x_2 \\ x_1 & 1 & x_3 \\ x_2 & x_3 & 1 \end{pmatrix}.$$

Damit eine solche Matrix positiv definit ist, müssen nach Satz 2.2.2 alle Hauptminoren echt positiv sein, es muss also gelten:

$$1 - x_1^2 > 0 \tag{4.1}$$
$$1 + 2x_1x_2x_3 - x_1^2 - x_2^2 - x_3^2 > 0 \tag{4.2}$$

Hieraus ergeben sich folgende notwendige Bedingung an die Parameter x_1, x_2, x_3:

$$-1 < x_1, x_2, x_3 < 1.$$

Möchte man nun eine Matrix erstellen mit der Nebenbedingung, dass $x_1 = 0$ gilt, so ist eine solchen Matrix positiv definit, falls x_2 und x_3 die Bedingung $x_2^2 + x_3^2 < 1$ erfüllen. Betrachtet man eine graphische Darstellung, so liegen alle Punkte im Kreis um 0 mit Radius 1 (Abbildung 4.1). Wenn man aber x_2 und x_3 mit $-1 < x_2, x_3 < 1$ beliebig vorgibt, so kann man x_1 wählen als $x_1 := x_2x_3$ und die resultierende Matrix ist positiv definit, denn es gilt $-1 < x_1 < 1$ und Bedingung 4.2 reduziert sich zu

$$(1 - x_3^2) \cdot (1 - x_2^2) > 0$$

und ist auch erfüllt. Dies zeigt, dass für eine positiv definite Matrix M mit $x_2^2 + x_3^2 \geq 1$, die Matrix durch Setzen des Parameters x_1 auf Null nicht positiv definit bleibt.

Es sind verschiedene Ansätze bekannt, Matrizen aus $SPR(n, H)$ zu generieren, von denen drei im Folgenden vorgestellt werden. Darauf folgt ein neuer Ansatz, bei dem die prämoralisierbaren Graphen die zentrale Rolle einnehmen.

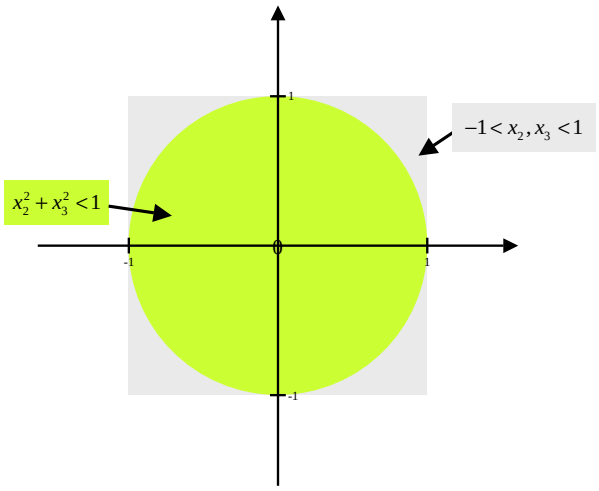

Abbildung 4.1: Darstellung des Definitionsbereich der Parameter x_2 und x_3 für symmetrische positiv definite Matrizen mit und ohne Nebenbedingungen an den Parameter x_1.

4.1.1 Ansatz 1: Diagonaldominante Matrizen

Die Menge der diagonaldominanten Matrizen ist in dieser Arbeit wie folgt definiert.

Definition 4.1.1 *Eine Matrix* $M = (m_{ij}) \in M(n, n, \mathbb{R})$ *heißt diagonaldominant, falls für alle* $i \in \{1, \ldots, n\}$ *gilt, dass*

$$m_{ii} > \sum_{\substack{j=1 \\ j \neq i}}^{n} |m_{ji}|.$$

Andere Definitionen von Diagonaldominanz fordern nur, dass der Betrag des Diagonalelementes echt größer ist als die Summe der Beträge der Nicht-Diagonalelemente. In dieser Arbeit wird allerdings das etwas stärkere Kriterium dafür verlangt, dass eine Matrix diagonaldominant ist. Mit dieser Definition gilt nun folgender bekannter Satz, dessen Beweis einen Spezialfall des Beweises des Gerschgorin-Theorems[29] darstellt und die Idee der Gerschgorin-Kreise benutzt.

Satz 4.1.2 *Die Menge der diagonaldominanten symmetrischen Matrizen ist in der Menge der positiv definiten symmetrischen Matrizen enthalten.*

Beweis: Da eine Matrix A positiv definit ist, genau dann wenn alle Eigenwerte positiv sind (siehe [23]), genügt es zu zeigen, dass für jeden Eigenwert λ ein i existiert mit

$$|\lambda - a_{ii}| \leq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|,$$

denn die rechte Seite der Ungleichung kann auf Grund der Diagonaldominanz durch a_{ii} abgeschätzt werden und somit folgt sofort $\lambda > 0$. Sei also $A = (a_{ij}) \in M(n, n, \mathbb{R})$ eine diagonaldominante, symmetrische Matrix und λ ein Eigenwert von A. Sei v ein Eigenvektor zum Eigenwert λ, das bedeutet es gilt $Av = \lambda v$. Man wählt nun einen Index i so, dass v_i vom Betrag maximal ist, das heißt für dieses i gilt $|v_i| = \max_j |v_j|$, also insbesondere $|v_i| > 0$ da $v \neq 0$. Somit erhält man

$$\lambda v_i = \sum_{j=1}^{n} a_{ij} v_j.$$

$$\Rightarrow \quad (\lambda - a_{ii}) \cdot v_i = \lambda v_i - a_{ii} v_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} v_j$$

Teilt man diese Gleichung durch v_i und geht zur Norm über, so ergibt sich

$$|\lambda - a_{ii}| \leq \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{|a_{ij}| \cdot |v_j|}{|v_i|}.$$

Da aber v_i vom Betrag her maximal gewählt worden ist, gilt $|v_j|/|v_i| \leq 1$ für alle $j \neq i$ und somit folgt die Aussage.

q.e.d.

Mit Satz 4.1.2 hat man nun eine einfache Möglichkeit, Elemente aus $SPR(n, H)$ zu generieren. Dazu erzeugt man sich eine beliebige Matrix $M = (m_{ij}) \in M(n, n, \mathbb{R})$, die den Graphen repräsentiert, und definiert dann für jedes i

$$m_{ii} := \sum_{j=1}^{n} \mid m_{ji} \mid + \epsilon_i$$

mit $\epsilon_i > 0$. Die so generierte Matrix ist diagonaldominant. Dies ist ein sehr einfaches und schnelles Verfahren, um positiv definite Matrizen mit Nebenbedingungen zu erstellen. Leider decken aber schon die diagonaldominanten Matrizen nur einen Teil aller positiv definiten Matrizen ab [57].

4.1.2 Ansatz 2: Hyper-inverse Wishart-Verteilung

Die hyper-inverse Wishart-Verteilung ist eine Verteilung auf der Menge $SPR(n, H)$ für einen ungerichteten Graphen H. Diese Verteilung wurde für zerlegbare Graphen H eingeführt von Dawid und Lauritzen [16]. Sie stellt eine Verallgemeinerung der inversen Wishart-Verteilung dar. Deshalb ist es sinnvoll, zuerst diese Klasse zu definieren.

Definition 4.1.3 *Sei $\delta > 0$ und $B = (b_{ij}) \in M(n, n, \mathbb{R})$ eine symmetrische positiv definite Matrix, das heißt $B \in SP(n)$. Für eine positiv definite Matrix Σ ist die Dichte f^{iw} der inversen Wishart-Verteilung gegeben durch*

$$f_n^{iw}(\Sigma|\delta, B) = h(\delta, n)\frac{|\Sigma|^{-(\delta+2n)/2}}{|B|^{-(\delta+n-1)/2}} \exp\left(-\frac{1}{2}tr(\Sigma^{-1}B)\right).$$

Die Normalisierungskonstante ist gegeben durch

$$h(\delta, n) = \frac{2^{-n(\delta+n-1)/2}}{\Gamma_n(\frac{1}{2}(\delta+n-1))},$$

wobei Γ_n die multivariate Gamma-Verteilung ist. Ist eine Matrix Σ invers Wishart-verteilt zu den Parametern δ und B, so schreibt man $\Sigma \sim IW(\delta, B)$.

Wenn man Daten aus der inversen Wishart-Verteilung erzeugt, so sind dies symmetrische positiv definite Matrizen. Ist eine Matrix Σ invers Wishart-verteilt, so gilt für $\Omega = \Sigma^{-1}$ dass diese Matrix Wishart-verteilt ist, also folgt $\Omega \in SP(n)$. Da $SPR(n, H) = SP(n)$, falls der Graph H vollständig ist, sind für Ω also keine zusätzlichen Bedingungen gegeben, das bedeutet es ist nicht gefordert, dass gewisse Elemente von Ω auf Null gesetzt werden. Man möchte nun aber diese Verteilung erweitern, so dass zum einen Eigenschaften der Wishart-Verteilung weiterhin gelten, andererseits aber auch Restriktionen durch beliebige Graphen gegeben werden können. Die folgenden Definitionen und Bermerkungen basieren auf einer Zusammenfassung der Resultate in Arbeiten von Roverato[55] und Carvalho[11]. Für die Definition der hyper-inversen Wishart-Verteilung nach Dawid und Lauritzen[16] und den daraus resultierenden Verallgemeinerungen muss ein ungerichteter Graph $H = (V, E)$ gegeben sein und man betrachtet eine Zerlegung dieses Graphen in die Primkomponenten und die zugehörigen Seperatoren.

Definition 4.1.4 *Sei $H = (V, E)$ ein ungerichteter Graph. Eine endliche Folge $(P_i = (V_i, E_i))_{i=1}^k$ von Teilgraphen von H und eine endliche Folge von vollständigen Teilgrahen $(S_i)_{i=2}^k$ sei definiert durch die folgenden Bedingungen.*

- *Für P_i existiert keine echte Zerlegung*
- $\bigcup_{i=1}^k V_i = V$
- $S_i = P_i \cap (\bigcup_{j=1}^{i-1} P_j)$

Man bezeichnet P_i als die Primkomponenten und S_i als Seperatoren. Eine Folge von Primkomponenten und Seperatoren wird als perfekt bezechnet, falls für alle $i \geq 2$ ein $j < i$ existiert mit

$$S_i \subset P_j.$$

Die Struktur eines Graphen ist durch die Angabe einer perfekten Folge von Primkomponenten und Seperatoren, die immer existiert, vollständig beschrieben. Anschaulich lässt sich diese Folge als eine Form einer Zerlegung des gesamten Graphen in kleinere, handlichere Strukturen interpretieren, vergleichbar mit der Primfaktorzerlegung einer Zahl. Möchte

man nun eine Matrix mit Nebenbedingungen erzeugen, wobei diese durch einen Graphen gegeben sind, so kann man iterativ die einzelnen Bereiche der Matrix besetzen, die zu den Primkomponenten und den Seperatoren des Graphen korrespondieren. Besonders einfach ist die Besetzung bei zerlegbaren Graphen, denn nach Definition ist bei zerlegbaren Graphen jede Primkomponente vollständig. Also muss die Matrix auf den Primkomponenten nur die positive Definitheit und die Symmetrie erfüllen, und solche Matrizen lassen sich mit Hilfe der (inversen) Wishart-Verteilung erzeugen. Die hyper-inverse Wishart-Verteilung basiert auf dieser Idee. Es ist eine Verteilung, die durch die Randverteilungen auf den Primkomponenten bestimmt ist.

Definition 4.1.5 *Sei $H = (V, E)$ ein zerlegbarer Graph, und P_i und S_i ein perfekte Folge von zugehörigen Primkomponenten und Seperatoren. Sei $\delta > 0$ und B eine positiv definite, symmetrische Matrix. Die hyper-inverse Wishart-Verteilung (HIW) ist die Verteilung, deren Randverteilung auf den Primkomponenten der inversen Wishart-Verteilung entspricht. Das bedeutet, eine Matrix Σ ist hyper-invers Wishart-verteilt zu den Parametern B und δ, falls $\Sigma_{P_iP_i} \sim IW(\delta, B_{P_iP_i})$ für alle Primkomponenten P_i. Die Dichte der HIW-Verteilung ist gegeben durch*

$$f_H(\Sigma|\delta, B) = \frac{\prod_{j=1}^{k} f^{iw}_{|P_i|}(\Sigma_{P_j,P_j}|\delta, B_{P_j,P_j})}{\prod_{j=2}^{k} f^{iw}_{|S_j|}(\Sigma_{S_j,S_j}|\delta, B_{S_j,S_j})}.$$

Ist eine Matrix Σ hyper-invers Wishart-verteilt zu einem Graphen H und den Parametern δ und B, so schreibt man $\Sigma \sim HIW_H(\delta, B)$.

Ansätze, um Daten aus einer hyper-inversen Wishart-Verteilung bei einem gegebenen zerlegbaren Graphen zu generieren, sind in Carvalho[11] aufgeführt. In Roverato[55] wird gezeigt, dass, falls $\Sigma \in HIW_H(\delta, B)$, so folgt $\Sigma^{-1} \in SPR(n, H)$. Man hat also für den Fall der zerlegbaren Graphen eine Verteilung definiert, deren Realisierungen (beziehungsweise die Inversen der Realisierungen) die gewünschten Nebenbedingungen erfüllen.

Man möchte aber auch nicht zerlegbare Graphen als Grundlage der Nebenbedingungen zulassen. Ziel ist es deshalb, Matrizen aus einer Verteilung zu erstellen, die für die vollständigen Primkomponenten der HIW entspricht, aber auch nicht vollständige Primkomponenten berücksichtigt. Als Beispiel für eine Realisierung eines solchen Vorhabens sei der Ansatz von Roverato[56] genannt. Dieser Ansatz wird auch bei der Validierung eines vorgestellten Netzwerkalgorithmus in einer Arbeit von Castelo[12] benutzt. Roverato beschreibt eine Methode, Daten aus einer Verallgemeinerung der hyper-inversen Wishart-Verteilung zu erzeugen, selbst wenn der zu Grunde liegende Graph nicht zerlegbar ist. Wie bei der HIW für zerlegbare Graphen wird für die Randverteilung auf den vollständigen Primkomponenten die inverse Wishart-Verteilung zu Grunde gelegt. Für die Matrix Σ_{PP}, die zu einer nicht vollständigen Primkomponente P korrespondiert, wird zunächst von einer inversen Wishart-Verteilung ausgegangen und die Wishart-Verteilung für $\Omega_{PP} = \Sigma^{-1}_{PP}$ ermittelt. Diese wird aber restringiert, so dass Realisierungen der Inversen die geforderten Nebenbedingungen erfüllen. Roverato beschreibt einen Ansatz, mit dem man Matrizen erzeugen kann, die aus der beschriebenen Verteilung stammen.

Hierzu startet man mit einem ungerichteten Graphen $H = (V, E)$, welcher die Restriktionen auf der Primkomponente beschreibt. Man gibt sich dann eine obere Dreiecksmatrix K^* vor, bei der die zu den Kanten aus H korrespondierenden Elemente, auch als freie Elemente bezeichnet, beliebig besetzt werden, und auf der Diagonalen nur positive Werte stehen dürfen. Eine solche Matrix K^* kann dann so zu einer Matrix K erweitert werden, dass $K^t \cdot K = Q \in SPR(n, H)$. Bei dem Ansatz wird beschrieben, dass die freien Elemente in der oberen Dreiecksmatrix voneinander unabhängig sind und deshalb für eine Realisierung einer solchen Matrix auch unabhängig erzeugt werden können. Die übrigen Einträge werden auf eindeutige Art und Weise besetzt und das Ergebnis ist eine positiv definite Matrix mit den geforderten Nebenbedingungen.

In dem Ansatz von Roverato wird also eine positiv definite Matrix mit Nebenbedingungen mit Hilfe einer oberen Dreiecksmatrix erzeugt. Auf ähnliche Weise wird in Kapitel 4.2 vorgegangen, allerdings wird hier die Anzahl der freien Elemente reduziert. Man hat dann den Vorteil, dass man die erzeugte Matrix K nicht mehr erweitern muss, sondern es reicht aus, die nicht freien Elemente auf Null zu setzen.

4.1.3 Ansatz 3: Optimierungsansätze

Eine mögliche Strategie zur Erzeugung von positiv definiten Matrizen liegt darin, mit der Fragestellung zu starten, zu einer vorgegebenen Matrix A eine positiv definite Matrix Ω zu finden, die bezüglich einer gegebenen Norm einen möglichst geringen Abstand besitzt. Für dieses Optimierungsproblem sind Algorithmen bekannt, beispielsweise für positiv semidefinite Matrizen [35]. Gibt man sich nun aber eine beliebige symmetrische Matrix A vor, die durch einen Graphen H gegebene Nebenbedingungen erfüllt, und betrachtet die durch einen solchen Algorithmus gefundene positiv definite Matrix $\hat{A}$, so erfüllt diese nicht mehr die geforderten Nebenbedingungen. Man muss also die Eigenschaft als zusätzliche Bedingung an die Matrix stellen. Dies ist bei einem *mathematischen Programm* möglich (siehe beispielsweise Bennett[6] für einen Überblick). Ein mathematisches Programm ist ein Problem, bei dem eine Funktion $f(s)$ minimiert werden soll, wobei man an s weitere Bedingungen stellen kann. Diese Nebenbedingungen lassen sich beschreiben durch

$$\begin{aligned} g(s) &\leq 0 \\ h(s) &= 0 \\ s &\in \Omega \end{aligned}$$

für gegebene Funktionen g,h und eine Menge Ω. Je nachdem, wie die Funktionen f,h und g sowie die Menge Ω definiert sind, unterscheidet man verschiedene Arten von mathematischen Programmen. Bei linearen Programmen ist beispielsweise sowohl die zu minimierende Funtion f also auch die die Nebenbedingungen beschreibenden Funtionen g und h linear. Und bei der semidefiniten Programmen wird der Ansatz der linearen Programme auf Matrizen übertragen. Hier enthält die Menge Ω insbesondere nur positiv semidefinite Matrizen.

Nun ist es möglich, auch das oben beschriebene Problem als mathematisches Programm zu formulieren. Dazu definiert man die Funktion f als $f(s) := \|A - s\|$ und stellt an s die weiteren Nebenbedingungen, dass s eine positiv definite Matrix ist, dass s die durch den Graphen H gegebenen Nebenbedingungen erfüllt und dass die Diagonale von s identisch 1 ist; die letzte Eigenschaft ist nötig, um das Resultat als partielle Korrelationsmatrix zu interpretieren (siehe Kapitel 4.4). Da viele Algorithmen bekannt sind, die mathematische Programme lösen, und dies auch ein aktueller Forschungsbereich ist, ist es mit der mathematischen Programmierung möglich, die gewünschten Matrizen zu generieren. Dies liegt zwar nicht im Fokus dieser Arbeit, kann aber als lohnender Startpunkt für weitere Untersuchungen angesehen werden.

4.2 Prämoralisierbare Graphen und positiv definite Matrizen

Dieser Abschnitt beschreibt eine weitere Möglichkeit, Elemente aus $SPR(n, H)$ zu erzeugen, falls der Graph H prämoralisierbar ist. Der Satz 4.2.2, der die Grundlagen für die weiteren Untersuchungen bildet, wird vorgestellt. Zudem werden Folgerungen aus diesem Satz gemacht, die Zusammenhänge zwischen der Moralisierung eines Graphen und einer Matrix Q der Form $Q = K^t \cdot K$ für eine obere Dreiecksmatrix K zeigen (siehe auch Abbildung 1.2). Insbesondere wird auch eine in der Arbeit von Dobra[18] aufgestellte Behauptung widerlegt. Für all diese Dinge bedarf es einiger Vorbereitungen. Diese beschäftigen sich mit der Repräsentierung von Graphen durch Matrizen und mit der Beschreibung von Verteilungsfunktionen durch graphische Strukturen.

Definition 4.2.1 *Sei P eine Verteilungsfunktion und $G = (V, \overrightarrow{E})$ ein gerichteter azyklischer Graph. P faktorisiert rekursiv über G, falls P eine Dichte p hat, die folgende Form erfüllt:*

$$p(x) = \prod_{v \in V} p(x_v | x_{P_G(v)})$$

Wenn eine Verteilungsfunktion rekursiv faktorisiert, so ist es möglich, Aussagen über Unabhängigkeitsstrukturen zu treffen, wie folgender Satz zeigt (Korrolar 3.23 aus [40]).

Satz 4.2.2 *Eine Verteilungsfunktion P faktorisiere rekursiv über einen gerichteten azyklischen Graphen $G = (V, \overrightarrow{E})$. Seien A,B und S unabhängige Teilmengen von V. Dann gilt:*

$$A \bowtie B | S[G^m_{An(A \cup B \cup S)}] \Rightarrow X_A \perp X_B | X_S.$$

Hierbei ist $An(A) := A \cup \{v \in V |$ es existiert ein Pfad von v nach $A\}$.

Man erhält als Folgerung:

Satz 4.2.3 *Eine Verteilungsfunktion P faktorisiere rekursiv über einen gerichteten azyklischen Graphen $G = (V, \overrightarrow{E})$ mit $V = \{1, \ldots, n\}$. Sei $H = (V, E^m)$ die Moralisierung von G. Sei $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ die zu P gehörende Präzessionsmatrix. Seien n_1,n_2 zwei Knoten aus V. Dann gilt:*

$$\{n_1, n_2\} \notin E^m \Rightarrow \omega_{n_1 n_2} = 0.$$

Beweis: Man möchte Satz 4.2.2 anwenden und zeigt deshalb:

$$\{n_1, n_2\} \notin E^m \Rightarrow \{n_1\} \bowtie \{n_2\} | (V \setminus \{n_1, n_2\})[H].$$

Die Umkehrung dieser Aussage wird gezeigt. Angenommen $\{n_1\} \bowtie \{n_2\} | (V \setminus \{n_1, n_2\})[H]$ gilt nicht. Dann gibt es einen Pfad zwischen n_1 und n_2, der nicht durch $V \setminus \{n_1, n_2\}$ läuft. Das heißt aber, dass dieser Pfad und somit der Graph die Kante $\{n_1, n_2\}$ enthalten muss. Somit gilt aber $\{n_1, n_2\} \in E^m$. Wendet man nun Satz 4.2.2 an mit $A = \{n_1\}$, $B = \{n_2\}$ und $S = V \setminus \{n_1, n_2\}$, dann gilt $G^m_{An(A \cup B \cup S)} = H$ und somit erhält man

$$\{n_1, n_2\} \notin E^m \Rightarrow X_{n_1} \perp X_{n_2} | X_{V \setminus \{n_1, n_2\}}$$

und mit

$$X_{n_1} \perp X_{n_2} | X_{V \setminus \{n_1, n_2\}} \Leftrightarrow \omega_{n_1 n_2} = 0.$$

folgt die Aussage.

q.e.d.

Mit diesen Vorbereitungen kann man nun das Theorem beweisen, welches die Grundlage des Simulationsalgorithmus aus dem nächsten Abschnitt liefert.

Theorem 4.2.4 *Sei $G = (V, \overrightarrow{E})$ ein DAG mit $V = \{1, \ldots, n\}$ und sei $H = (V, E^m)$ die Moralisierung von G. Sei $K_G = (k_{ij}) \in M(n, n, \mathbb{R})$ eine Matrix, die G repräsentiert. Sei eine Diagonalmatrix $D = (d_{ij}) \in M(n, n, \mathbb{R})$ mit $d_{ii} \neq 0$ für alle i gegeben. Definiere die Matrix $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ durch*

$$\Omega := (D - K_G)^t \cdot (D - K_G).$$

Dann ist Ω eine positiv definite Matrix und es gilt für alle Knoten $n_1, n_2 \in V$:

$$\{n_1, n_2\} \notin E^m \Rightarrow \omega_{n_1 n_2} = 0.$$

Beweis: Man möchte Satz 4.2.3 anwenden und dazu eine Verteilung konstruieren, die rekursiv über G faktorisiert und deren Präzessionsmatrix Ω ist. Man definiert die n-dimensionale Zufallsvariable x durch folgenden Zusammenhang:

$$(D - K_G) \cdot x = \epsilon \qquad \epsilon \sim N_n(0, Id)$$

Da $(D - K_G)$ eine obere Dreiecksmatrix mit vollem Rang ist, folgt nach Satz 2.2.10, dass x normalverteilt mit Präzessionsmatrix Ω ist. Es muss noch gezeigt werden, dass die Dichte

$p(x)$ rekursiv über G faktorisiert. Durch die obere Definitionsgleichung von x wird auch eine Reihenfolge der Elemente von x impliziert. Die Dichte von x lässt sich immer schreiben als

$$p(x) = \prod_{i=1}^{p} p(x_i | x_{i+1}, \ldots x_p).$$

Die obere Matrix K_G repräsentiert aber den Graphen G, so dass $p(x_i|x_{i+1}, \ldots x_p) = p(x_i|x_{P_G(i)})$ für alle i. Damit faktorisiert die Dichte von x rekursiv über G.

q.e.d.

Wie schon in Kapitel 2 bei der Beschreibung des Dobra-Algorithmus angemerkt, gilt die Rückrichtung von Theorem 4.2.4 in der Allgemeinheit nicht. Dies ist auch im Zusammenhang mit dem von Dobra vorgestellten Algorithmus sehr kritisch zu sehen, denn es kann durchaus vorkommen, dass Elemente der Matrix Ω Null sind, aber trotzdem eine Kante im zugehörigen Graphen existiert. Betrachtet man für die Untersuchungen also nur den von Dobra konstruierten Graphen, so enthält dieser Kanten zwischen Genen, die in der zu Grunde liegenden Verteilung gar nicht existieren. Damit sind natürlich falsche Schlüsse möglich. Dazu wird nun ein Beispiel konstruiert.

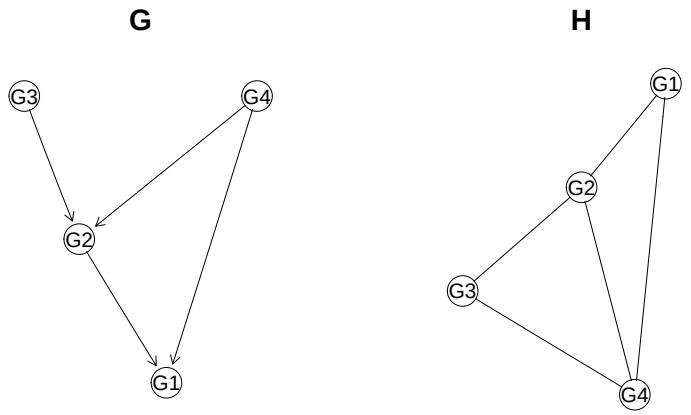

Abbildung 4.2: Gegenbeispiel 1 zur Rückrichtung von Theorem 4.2.4. G ist eine DAG und H die zugehörige Moralisierung.

Beispiel 1: Man betrachtet den DAG $G = (V_G, \overrightarrow{E_G})$ (siehe Abbildung 4.2) mit

$$\begin{aligned} V_G &= \{G1, G2, G3, G4\} \\ \overrightarrow{E_G} &= \{(G3, G2), (G2, G1), (G4, G1), (G4, G2)\} \end{aligned}$$

Die Moralisierung von G ist gegeben durch $H = (V_H, E_H)$ mit

$$\begin{aligned} V_H &= \{G1, G2, G3, G4\} \\ E_H &= \{\{G1, G2\}, \{G1, G4\}, \{G4, G2\}, \{G3, G2\}, \{G3, G4\}\} \end{aligned}$$

wobei die Kante $\{G4, G3\}$ durch Moralisierung entsteht. Die folgende Matrix K_G ist die kanonische Repräsentation von G:

```
   G1 G2 G3 G4
G1  0  1  0  1
G2  0  0  1  1
G3  0  0  0  0
G4  0  0  0  0
```

Wählt man nun $D = Id_4$, so ergibt sich für $\Omega = (Id_4 - K_G)^t \cdot (Id_4 - K_G)^t$

```
   G1 G2 G3 G4
G1  1 -1  0 -1
G2 -1  2 -1  0
G3  0 -1  2  1
G4 -1  0  1  3
```

Die Einträge ω_{13} und ω_{24} sind Null. Es gilt aber $\{G4, G2\} \in E_H$, somit existiert eine Kante in H, für die der entsprechende Eintrag in Ω Null ist. Folglich ist die Rückrichtung von Theorem 4.2.4 nicht allgemein gültig.

In Beispiel 1 ist die Kante $\{G4, G2\}$, für die der entsprechende Eintrag in Ω Null ist, nicht durch Moralisierung entstanden, denn $(G4, G2) \in \overrightarrow{E_G}$. Um zu zeigen, dass auch der Fall auftreten kann, in dem eine Kante durch Moralisierung entsteht und trotzdem der entsprechende Eintrag in der Matrix Ω Null ist, wird nun ein zweites Beispiel konstruiert.

Beispiel 2: In diesem Beispiel betrachtet man den DAG $G = (V_G, E_G)$ (Abbildung 4.3) mit

$$\begin{aligned} V_G &= \{G1, G2, G3, G4\} \\ \overrightarrow{E_G} &= \{(G3, G1), (G3, G2), (G4, G1), (G4, G2)\} \end{aligned}$$

Die Moralisierung von G ist der Graph $H = (V_H, E_H)$ mit

$$\begin{aligned} V_H &= \{G1, G2, G3, G4\} \\ E_H &= \{\{G1, G3\}, \{G1, G4\}, \{G4, G2\}, \{G3, G2\}, \{G3, G4\}\} \end{aligned}$$

Die G repräsentierende Matrix K_G sei

```
   G1 G2  G3   G4
G1  0  0 0.5 -0.5
G2  0  0 0.5  0.5
G3  0  0 0.0  0.0
G4  0  0 0.0  0.0
```

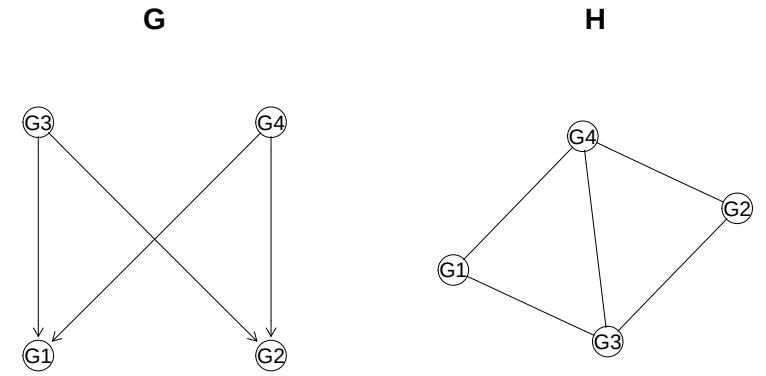

Abbildung 4.3: Gegenbeispiel 2 zur Rückrichtung von Theroem 4.2.4. G ist ein DAG und H die zugehörige Moralisierung.

Wählt man nun $D = Id_4$ so ergibt sich für $\Omega = (Id_4 - K_G)^t \cdot (Id_4 - K_G)^t$

```
     G1   G2   G3   G4
G1  1.0  0.0 -0.5  0.5
G2  0.0  1.0 -0.5 -0.5
G3 -0.5 -0.5  1.5  0.0
G4  0.5 -0.5  0.0  1.5
```

Es gilt nun $\{G3, G4\} \in E_H$ aber $\omega_{34} = \omega_{43} = 0$. Die Kante $\{G3, G4\}$ ist aber durch Moralisierung entstanden.

In Beispiel 2 sind sowohl negative als auch positive Einträge in der den Graphen G repräsentierenden Matrix K_G vorhanden. Man kann beweisen, dass der in Beispiel 2 gezeigte Fall nicht auftreten kann, wenn sich in der Matrix K_G nur positive Einträge befinden. Das bedeutet, falls Kanten im moralisierten Graphen H vorhanden sind, und die Matrix Ω trotzdem an dieser Stelle eine Null hat, so muss die Kante auch schon im DAG G vorhanden gewesen sein.

Satz 4.2.5 *Sei $G = (V, \overrightarrow{E_G})$ ein DAG und sei $V = \{1, \ldots, n\}$. Sei $K = (k_{ij}) \in M(n, n, \mathbb{R})$ eine den Graphen G repräsentierende Matrix mit $k_{ij} \geq 0$ für alle $j > i$. Sei $H = (V, E^m)$ die Moralisierung von G. Sei D eine Diagonalmatrix mit $d_{ii} \neq 0$ für alle i. Sei $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ definiert durch $\Omega = (D - K)^t \cdot (D - K)$. Dann gilt: Für alle Kanten $\{i, j\} \in E^m$ mit $\omega_{ij} = 0$ gilt $(i, j) \in \overrightarrow{E_G}$ oder $(j, i) \in \overrightarrow{E_G}$.*

Beweis: Sei $\{i, j\} \in E^m$ und $\omega_{ij} = 0$. Sei o.B.d.A. $j < i$. Nun erfolgt der Beweis durch Widerspruch. Angenommen $(i, j) \notin E$ und $(j, i) \notin E$. Da K den Graphen G repräsentiert, gilt somit $k_{ji} = k_{ij} = 0$. Da aber $\{i, j\} \in E^m$ ist die Kante durch Moralisieren entstanden,

das heißt es existiert ein $a \in V$ mit $(i,a) \in \overrightarrow{E_G}$ und $(j,a) \in \overrightarrow{E_G}$. Das bedeutet aber für die Matrix K, dass $k_{ai} > 0$ und $k_{aj} > 0$. Sei $M = (m_{ij}) \in M(n,n,\mathbb{R})$ mit $M = (D-K)$. Nach Voraussetzung gilt dann $m_{rr} \neq 0$ für alle r und $m_{st} = 0$ für alle $s > t$, und es gilt $m_{st} \leq 0$ für alle $s < t$. Es gilt zudem $m_{ji} = 0$ und $\omega_{ij} = 0$, also

$$\begin{aligned} 0 &= \omega_{ij} \\ &= \sum_{r=1}^{j} m_{ri} \cdot m_{rj} \\ &= m_{ji} \cdot m_{jj} + m_{ai} \cdot m_{aj} + \sum_{\substack{r=1 \\ r \neq a}}^{j-1} m_{ri} \cdot m_{rj} \\ &= \underbrace{m_{ai} \cdot m_{aj}}_{>0} + \underbrace{\sum_{\substack{r=1 \\ r \neq a}}^{j-1} m_{ri} \cdot m_{rj}}_{\geq 0} \end{aligned}$$

Der erste Summand ist größer als Null, da sowohl $m_{ai} < 0$ als auch $m_{aj} < 0$ sind (weil $k_{ai} > 0$ und $k_{aj} > 0$). Der zweite Summand ist größer oder gleich Null, weil dort über alle $r < j < i$ summiert wird, und für diese Einträge gilt $m_{ri} \leq 0$ und $m_{rj} \leq 0$. Man erhält somit einen Widerspruch.

q.e.d.

Wie gezeigt ist die Umkehrung von Theorem 4.2.4 allgemein nicht gültig. Möchte man, dass die Umkehrung auch gilt, so müssen zusätzliche Forderungen an die Matrizen K und D gestellt werden. Hinreichende Forderungen liefert der folgende Satz.

Satz 4.2.6 *Sei $G = (V, \overrightarrow{E_G})$ ein DAG mit $|V| = n$. Sei H die Moralisierung von G. Sei $K = (k_{ij}) \in M(n,n,\mathbb{R})$ eine Matrix, die G repräsentiert mit $k_{ij} \leq 0$ für alle i und j. Sei $D = (d_{ij}) \in M(n,n,\mathbb{R})$ eine Diagonalmatrix mit $d_{ii} > 0$ für alle i. Definiere die Matrix $\Omega = (\omega_{ij}) \in M(n,n,\mathbb{R})$ durch*

$$\Omega := (D-K)^t \cdot (D-K)^t.$$

Dann ist Ω eine positiv definite Matrix und es gilt für alle Knoten n_1,n_2 aus H:

$$\{n_1, n_2\} \notin E^m \Leftrightarrow \omega_{n_1 n_2} = 0.$$

Beweis: Die Hinrichtung folgt aus Satz 4.2.4. Für die Rückrichtung zeigt man die Negation der Aussage, das heißt man möchte für n_1 und n_2 mit $\{n_1, n_2\} \in E^m$ und $n_2 < n_1$ zeigen, dass $\omega_{n_1 n_2} \neq 0$. Definiere hierzu $M = (m_{ij}) \in M(n,n,\mathbb{R})$ durch $M = (D-K)$. Nach Voraussetzung gilt für alle s,t dass $m_{st} \geq 0$ und $m_{tt} > 0$. Man unterscheidet nun zwei Fälle.

Fall 1: Es gilt $(n_1, n_2) \in \overrightarrow{E_G}$. Dann gilt $m_{n_2 n_1} > 0$ und $m_{n_2 n_2} > 0$. Somit folgt

$$\omega_{n_1 n_2} = \sum_{r \leq n_2} m_{r n_1} \cdot m_{r n_2} = \underbrace{m_{n_2 n_1} \cdot m_{n_2 n_2}}_{>0} + \underbrace{\sum_{r \leq (n_2 - 1)} m_{r n_1} \cdot m_{r n_2}}_{\geq 0} > 0$$

Fall 2: Es gilt $(n_1, n_2) \notin \overrightarrow{E_G}$. Dann entsteht die Kante durch Moralisierung, das heißt es existiert ein $a \in V$ mit $a < n_2$ sowie $(n_1, a) \in \overrightarrow{E_G}$ und $(n_2, a) \in \overrightarrow{E_G}$. Für die Elemente der Matrix M gilt dann aber $m_{a n_1} > 0$ und $m_{a n_2} > 0$. Somit erhält man

$$\omega_{n_1 n_2} = \sum_{r \leq n_2} m_{r n_1} \cdot m_{r n_2} = \underbrace{m_{a n_1} \cdot m_{a n_2}}_{>0} + \underbrace{\sum_{\substack{r \leq n_2 \\ r \neq a}} m_{r n_1} \cdot m_{r n_2}}_{\geq 0} > 0$$

q.e.d.

4.2.1 Der PddP Algorithmus - Positiv definit durch Prämoralisierung

Mit Hilfe von Satz 4.2.4 wird nun ein neues Verfahren vorgestellt, welches es erlaubt, positiv definite Matrizen mit Nebenbedingungen zu erzeugen, wobei die Nebenbedingungen durch einen prämoralisierbaren Graphen H mit $n = |V|$ gegeben sind. Sei also ein ungerichteter prämoralisierbarer Graph H gegeben, und es soll eine Matrix $\Omega \in SPR(n, H)$ erzeugt werden. Sei $K_G = (k_{ij}) \in M(n, n, \mathbb{R})$ eine Matrix, die eine Prämoralisierung G von H repräsentiert. Sei $D = (d_{ij}) \in M(n, n, \mathbb{R})$ eine Diagonalmatrix mit $d_{ii} \neq 0$ für alle i. Wenn man nun die Matrix Ω definiert als

$$\Omega := (D - K_G)^t \cdot (D - K_G)^t,$$

so gilt nach Satz 4.2.4 dass $\Omega \in SPR(n, H)$. Es ergibt sich somit folgender Algorithmus, um für prämoralisierbare Graphen H Elemente aus $SPR(n, H)$ zu erzeugen.

Algorithmus PddP

1. Erzeuge eine Prämoralisierung G von H (beispielsweise mit Algorithmus I oder Algorithmus II aus Kapitel 3).
2. Erzeuge eine den Graphen G repräsentierende Matrix K_G. Dabei werden die Knoten des Graphen G durch eine Permutation π so umgeordnet, dass K_G eine obere Dreiecksmatrix ist.
3. Erzeuge eine Diagonalmatrix $D = (d_{ij}) \in M(n, n, \mathbb{R})$ mit $d_{ii} \neq 0$.
4. Definiere $\Omega := (D - K_G)^t \cdot (D - K_G)^t$.

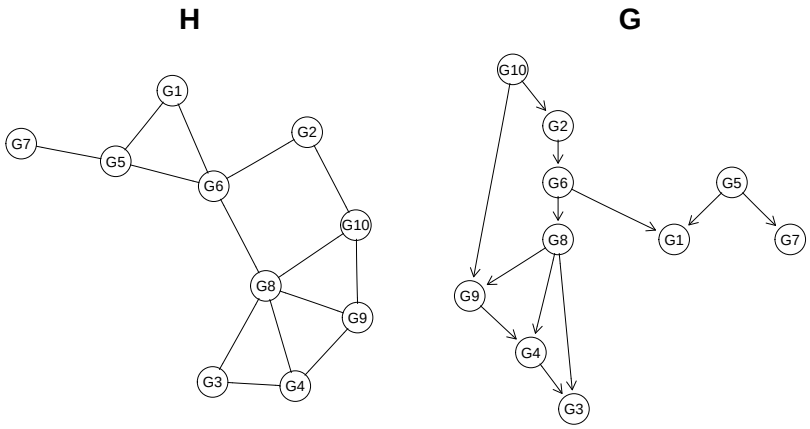

Abbildung 4.4: Beispiel für den PddP Algorithmus: Der vorgegebene prämoralisierbare Graph H und eine mögliche Prämoralisierung G

5. Wende π^{-1} auf die Zeilen und Spalten der Matrix Ω an, um die ursprüngliche Reihenfolge der Knoten zu erhalten.

Der PddP Algorithmus wird nun an einem Beispiel demonstriert. Gegeben ist ein prämoralisierbarer Graph $H = (V, E)$. Abbildung 4.4 zeigt H und eine Prämoralisierung G, erzeugt mit Hilfe von Algorithmus II. Dieser Graph wird von der folgenden Matrix K repräsentiert, wobei eine Permutation π der Knoten benutzt worden ist, um die Matrix als obere Dreiecksmatrix darstellen zu können.

	G3	G4	G9	G8	G1	G6	G7	G2	G10	G5
G3	0	0.86	0.00	0.47	0	0.00	0	0.00	0.00	0.00
G4	0	0.00	0.14	0.79	0	0.00	0	0.00	0.00	0.00
G9	0	0.00	0.00	0.32	0	0.00	0	0.00	0.63	0.00
G8	0	0.00	0.00	0.00	0	0.96	0	0.00	0.00	0.00
G1	0	0.00	0.00	0.00	0	0.19	0	0.00	0.00	0.68
G6	0	0.00	0.00	0.00	0	0.00	0	0.37	0.00	0.00
G7	0	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.23
G2	0	0.00	0.00	0.00	0	0.00	0	0.00	0.79	0.00
G10	0	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
G5	0	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00

Nun wird die Matrix Q erzeugt als $\Omega = (Id_{10} - K)^t \cdot (Id_{10} - K)$. Wendet man noch die Permutation π auf die Zeilen und Spalten der Matrix an, so ergibt sich für Ω

	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10
G1	1.00	0.00	0.00	0.00	-0.68	-0.19	0.00	0.00	0.00	0.00
G2	0.00	1.13	0.00	0.00	0.00	-0.37	0.00	0.00	0.00	-0.79
G3	0.00	0.00	1.00	-0.86	0.00	0.00	0.00	-0.47	0.00	0.00
G4	0.00	0.00	-0.86	1.73	0.00	0.00	0.00	-0.39	-0.14	0.00
G5	-0.68	0.00	0.00	0.00	1.52	0.13	-0.23	0.00	0.00	0.00
G6	-0.19	-0.37	0.00	0.00	0.13	1.95	0.00	-0.96	0.00	0.00
G7	0.00	0.00	0.00	0.00	-0.23	0.00	1.00	0.00	0.00	0.00
G8	0.00	0.00	-0.47	-0.39	0.00	-0.96	0.00	1.95	-0.21	0.20
G9	0.00	0.00	0.00	-0.14	0.00	0.00	0.00	-0.21	1.02	-0.63
G10	0.00	-0.79	0.00	0.00	0.00	0.00	0.00	0.20	-0.63	2.02

Die erzeugte Matrix Ω ist symmetrisch und positiv definit. Nach Satz 4.2.4 gilt für diese Matrix, dass dort, wo der Graph H keine Kante besitzt, der entsprechende Eintrag in Ω gleich Null ist. Somit gilt $\Omega \in SPR(n, H)$. Die in diesem Beispiel erzeugte Matrix ist nicht diagonaldominant. Daran kann man erkennen, dass mit dem PddP Algorithmus andere Matrizen erzeugt werden als beim Ansatz mit diagonaldominanten Matrizen. Weitere Beispiele in späteren Simulationsstudien werden dies auch belegen.

Der PddP Algorithmus zum Erzeugen der Matrizen aus der Menge $SPR(n, H)$ liefert Freiheiten in der Wahl der repräsentierenden Matrix für den Graphen G und in der Wahl der Diagonalmatrix D. Möchte man beispielsweise die Eigenschaft erfüllt wissen, dass ein Eintrag aus Ω genau dann nicht Null ist, wenn eine entsprechende Kante in H existiert, so wird in Satz 4.2.6 gezeigt, dass dies möglich ist, falls D nur positive Diagonalelemente und K nur nicht positive Elemente besitzt. Dies ist also eine erste mögliche Besetzungsstrategie. Ein anderer Ansatz, bei dem der Abstand zu einer vorgegebenen Matrix minimiert werden soll, wird im folgenden Abschnitt motiviert und erläutert.

4.3 Nutzung von positiv definiten Matrizen mit Nebenbedingungen - Validierung von Netzwerkalgorithmen auf Microarray-Daten

In diesem Abschnitt wird eine Motivation gegeben, warum es wichtig ist, Netzwerkalgorithmen zu validieren, die Graphen aus Microarray-Daten schätzen, und wie positiv definite Matrizen mit Nebenbedingungen für diese Validierung genutzt werden können. Dieser Abschnitt zeigt auch, dass diagonaldominante Matrizen für Validierungsstudien nachteilig sein können.

In Kapitel 2 sind zwei Algorithmen vorgestellt worden, mit deren Hilfe Netzwerke beziehungsweise Graphen aus Microarray-Daten erstellt worden sind. Beide Algorithmen versuchen, unter der Annahme einer multivariaten Normalverteilung $N(\mu, \Sigma)$ für die Expressionsdaten die Elemente der Präzessionsmatrix $\Omega = \Sigma^{-1}$ zu extrahieren, die ungleich Null

sind. Damit ist es möglich, die Genpaare G_1, G_2 zu finden, für die eine Korrelation gegeben alle anderen Gene existiert. Man erhofft sich, dass diese Gene direkt interagieren. Da Microarray-Daten sehr viele Parameter (Gene) aber nur wenige Beobachtungen (Chip-Hybridisierungen) haben, enthalten die Algorithmen, die Netzwerke aus Expressionsdaten generieren, häufig ein heuristisches Element. So wird beispielsweise im Ansatz von Dobra [18] eine Vorwärts- und Rückwärtssuche benutzt, um die Parameter für die linearen Modelle zu finden (siehe Kapitel 2). In den Artikeln, die solche Algorithmen vorstellen, finden sich höchstens asymptotische Beweise, beispielsweise in der Arbeit von Meinshausen[50], dass das Verfahren funktioniert. Diese Beweise haben für konkrete Fragestellungen allerdings wenig Relevanz.

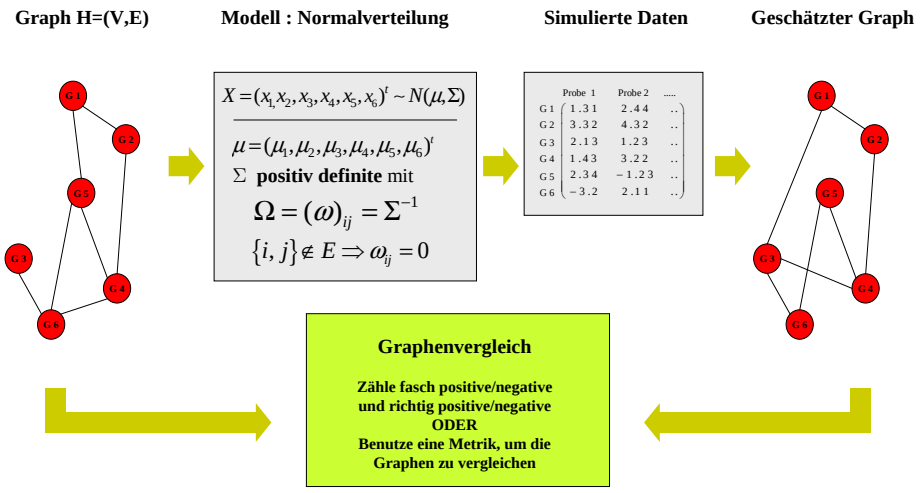

Abbildung 4.5: Ein möglicher Simulationsansatz zum Validieren eines Algorithmus, der Interaktionsnetzwerke aus Genexpressionsdaten erstellt.

Die Validierung eines entwickelten Algorithmus muss also darüber erfolgen, dass der Algorithmus an Daten eingesetzt wird und seine geschätzten Resultate mit der Wahrheit verglichen werden. Leider gibt es in der Biologie keine Datensätze, bei denen die komplette Wahrheit bekannt ist. Deshalb bietet sich eine Simulationsstudie an, um die Funktionsweise der Algorithmen zu testen. Durch simulierte Daten ist es möglich, sich die Wahrheit vorzugeben und somit kann überprüft werden, ob diese auch durch den Algorithmus gefunden wird.

Einen möglichen Ansatz für eine Simulationsstudie, um einen Algorithmus zum Schätzen eines Expressionsnetzwerkes zu validieren, zeigt Abbildung 4.5. Hier werden die Daten aus einer multivariaten Normalverteilung erzeugt. Dies kann für den zu untersuchenden

Algorithmus vorteilhaft sein, da viele Algorihmen, die Netzwerke aus Microarray-Daten schätzen, von der Normalverteilungsannahme ausgehen [18, 64, 63, 50]. Man ist somit in einem für den Algorithmus günstigen Fall. Andere Simulationsansätze sind denkbar, in denen die Daten nicht aus einer multivariaten Normalverteilung erzeugt werden, um damit zu testen, wie gut die Resultate des Algorithmus sind, falls die Annahmen nicht erfüllt sind. Eine solche Studie soll hier aber nicht betrachtet werden.

Aus Abbildung 4.5 erkennt man, dass das Erzeugen einer Matrix mit Nebenbedingungen eine zentrale Rolle bei der Durchführung einer Simulation spielt. Es müssen Daten aus einer multivariaten Normalverteilung $N(\mu, \Sigma)$ erzeugt werden, wobei die Inverse der Kovarianzmatrix aus $SPR(H, n)$ stammen soll.

Der naheliegendste Ansatz zur Validierung eines Netzwerkalgorithmus durch Simulationsstudien besteht darin, die Sensitivität und Spezifität zu bestimmen, indem man beim vom Algorithmus geschätzten Graphen $\tilde{H} = (V, \tilde{E})$ die gefundenen beziehungsweise nicht gefundenen Kanten mit den Kanten des vorgegebenen Graphen $H = (V, E)$ vergleicht. Wählt man einen solchen Ansatz, so gehen hierbei nicht die vorgegebenen partiellen Korrelationswerte ein. Um sicher zu stellen, dass alle Kanten die gleiche Wahrscheinlichkeit haben, gefunden zu werden, ist es somit sinnvoll, dass die vorgegebenen partiellen Korrelationen nahe 1 beziehungsweise -1 liegen. Für die zu Σ inverse Matrix Ω ist es demnach wünschenswert, dass folgende zwei Bedingungen gelten.

1. $\Omega \in SPR(H, n)$
2. Sei $\bar{\Omega} = (\bar{\omega}_{ij}) \in M(n, n, \mathbb{R})$ die partielle Korrelationsmatrix, das bedeutet $\bar{\Omega} := f(\Omega)$ wobei f die Funktion aus Definition 2.2.5 ist. Es soll dann gelten $|\bar{\omega}_{ij}| \approx 1$ für alle i, j mit $\bar{\omega}_{ij} \neq 0$.

Betrachtet man die diagonaldominanten Matrizen, so stellt man fest, dass die zweite Bedingung mit diesem Ansatz allgemein nicht erfüllt werden kann. Für diagonaldominante Matrizen sind die Diagonalelemente vom Betrag echt größer als die Spaltensumme, das bedeutet es ergibt sich für $\bar{\omega}_{ij}$

$$
\begin{aligned}
|\bar{\omega}_{ij}| &< \frac{|\omega_{ij}|}{\sqrt{\sum_{\substack{k=1\\k\neq i}}^{n} |\omega_{ki}| \cdot \sum_{\substack{k=1\\k\neq j}}^{n} |\omega_{kj}|}} \\
&= \frac{|\omega_{ij}|}{\sqrt{(|\omega_{ji}| + \sum_{\substack{k=1\\k\neq i,j}}^{n} |\omega_{ki}|) \cdot (|\omega_{ij}| + \sum_{\substack{k=1\\k\neq j,i}}^{n} |\omega_{kj}|)}} \\
&= \frac{|\omega_{ij}|}{\sqrt{\omega_{ij}^2 + |\omega_{ij}| \cdot \sum_{\substack{k=1\\k\neq i,j}}^{n} |\omega_{ki}| + |\omega_{ij}| \cdot \sum_{\substack{k=1\\k\neq j,i}}^{n} |\omega_{kj}| + \sum_{\substack{k=1\\k\neq i,j}}^{n} |\omega_{ki}| \cdot \sum_{\substack{k=1\\k\neq j,i}}^{n} |\omega_{kj}|}}
\end{aligned}
$$

Gibt es nun Knoten mit relativ vielen Nachbarn – gibt es also in der erzeugten Präzessionsmatrix eine Spalte mit vielen Einträgen ungleich Null – so wird die partielle Korrelation klein sein. Betrachtet man Realisierungen der hyper-inversen Wishart-Verteilung, so zeigt

sich in ersten Simulationsstudien schon für zerlegbare Graphen, dass die erzeugten Graphen auch nicht die Bedingung 2 erfüllen.

4.4 Besetzung der Elemente der Diagonalmatrix und der repräsentierenden Matrix

In diesem Abschnitt wird die Eigenschaft 2 aus dem letzten Abschnitt mit Hilfe einer Matrix A präzisiert. Dann wird eine Funktion OP_A definiert, die für eine beliebige Besetzung der Elemente der Diagonalmatrix D und der oberen Dreiecksmatrix K_G aus dem PddP-Algorithmus angibt, wie groß der Abstand bezüglich der Frobeniusnorm zwischen A und der Matrix $(D - K_G)^t \cdot (D - K_G)$ ist. Hierbei ist G eine Prämoralisierung des die Matrix A repräsentierenden Graphen. Die Funktion OP_A wird dann mit Hilfe der zwei in Kapitel 2 eingeführten Verfahren optimiert, wobei sich das BFGS-Verfahren als besser erweist. Der gesamte Vorgang wird schließlich in dem Algorithmus A zusammengefasst, wobei hier auch eine Einbindung von partiellen Varianzen möglich ist.

Sei $A = (a_{ij}) \in M(n, n, \mathbb{R})$ eine Matrix mit

$$\begin{aligned} a_{ii} &= 1 \quad \text{für alle } i \\ a_{ij} &\in \{1, -1\} \quad \text{falls } \{i, j\} \in E \\ a_{ij} &= 0 \quad \text{falls } \{i, j\} \notin E \end{aligned}$$

Eigenschaft 2 des letzten Abschnittes kann man nun auch wie folgt definieren.

2a. Sei $\bar{\Omega} = (\bar{\omega}_{ij}) \in M(n, n, \mathbb{R})$ die partielle Korrelationsmatrix, das bedeutet $\bar{\Omega} := f(\Omega)$, wobei f die Funktion aus Definition 2.2.5 ist. Es soll dann gelten $\|A - \bar{\Omega}\| \approx 0$.

Hierbei sei als gegebene Norm die Frobeniusnorm gewählt, denn diese spiegelt die Tatsache wider, dass jedes einzelne Element der Matrix einen möglichst geringen Abstand zu den vorgegebenen Elementen haben soll. Andere Normen beziehen sich nur auf den Abstand einzelner Elemente (Maximumsnorm) oder einzelner Spalten der Matrix. Zudem hat sich in den folgenden Studien zur Optimierung gezeigt, dass die Frobeniusnorm bessere Resultate liefert als beispielsweise die Maximumsnorm, da die zu optimierende Funktion *glatter* ist und nicht so starke Sprünge aufweist.

Man kann die Eigenschaft 2a nun auch so erweitern, dass man sich eine beliebige Matrix A vorgibt. Gegeben sei dann eine symmetrische aber nicht notwendigerweise positiv definite Matrix $A = (a_{ij}) \in M(n, n, \mathbb{R})$. Für diese Matrix gilt $a_{ii} = 1$ für alle i sowie $| a_{ij} | \leq 1$ für alle i und alle j. Zudem repräsentiert die Matrix A einen ungerichteten Graphen $H = (V_H, E_H)$. Es soll nun eine Matrix $\Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ mit folgenden Bedingungen erzeugt werden:

1. $\Omega \in SPR(H, n)$

2. $\|\Omega - \tilde{A}\|$ soll minimal sein, wobei $\tilde{A} = (\tilde{a}_{ij}) \in M(n, n, \mathbb{R})$ definiert ist als $\tilde{a}_{ii} = 1$ für alle i und $\tilde{a}_{ij} = -a_{ij}$ für alle $i \neq j$.

3. $\omega_{ii} = 1$ für alle i.

Es gilt also für die Matrix $\tilde{A}$, dass $\tilde{A} = g(A)$, wobei g definiert ist wie in Definition 2.2.8. Die Unterscheidung zwischen A und $\tilde{A}$ liegt darin begründet, dass man die vorgegebene Matrix A als eine Matrix interpretieren möchte, die die partiellen Korrelationen der zu erzeugenden multivariaten Normalverteilung $N(\mu, \Sigma)$ bestimmt. Für die spätere Simulation von Daten aus $N(0, \Sigma)$ ist es besser, eine Matrix zu erzeugen, die als Normierung einer inversen Kovarianzmatrix interpretiert werden kann. Nun unterscheiden sich die normierte inverse Kovarianzmatrix und die Matrix mit den partiellen Korrelationen aber nur durch das Vorzeichen der Nicht-Diagonalelemente. Hat man also eine Matrix Ω erstellt mit $\|\tilde{A} - \Omega\|$ klein, und interpretiert man dieses Ω als $f(\Sigma^{-1})$ einer multivariaten Normalverteilung $N(\mu, \Sigma)$, also als normierte inverse Kovarianzmatrix, so gilt für die zugehörige partielle Korrelationsmatrix $\tilde{\Omega} = g(\Omega)$, dass $\|A - \tilde{\Omega}\|$ ebenfalls klein ist.

Bei dem Optimierungsverfahren wird der Abstand zu einer vorgegebenen Matrix minimiert, die als Korrelationsmatrix interpretiert wird, was einen Vorteil bietet, da man die Korrelation beziehungsweise partielle Korrelation vorgeben möchte. Würde man eine Kovarianzmatrix vorgeben, wäre am Ende aber doch an den Korrelationen interessiert, so kann dies zu Problemen führen. Angenommen, man gibt sich eine Kovarianzmatrix A vor und erzeugt eine Matrix Ω, die bezüglich der Frobeniusnorm einen kleinen Abstand zur vorgegebenen Matrix A hat. Berechnet man dann die Korrelationen mit Hilfe von Ω und vergleicht diese mit den Korrelationen der Matrix A, so werden die Unterschiede dadurch unter Umständen groß, weil zu den Fehlern, die man durch die erzeugten Kovarianzen erhält, noch die Fehler hinzu kommen, die durch die erzeugten Varianzen entstehen.

Im Folgenden wird die Funktion definiert, welche dann durch die zwei Verfahren optimiert werden soll, die in Kapitel 2 beschrieben worden sind. Das Ziel ist die Erzeugung einer Matrix Ω, die die obigen Bedingungen 1-3 erfüllt. Sei also $G = (V_G, E_G)$ eine Prämoralisierung des oben eingeführten Graphen H, sei $n = |V_G|$ und $e = |E_G|$. Man definiert die kanonische bijektive Abbildung

$$\begin{aligned} \mathcal{K} : \mathbb{R}^{n^2} &\rightarrow M(n, n, \mathbb{R}) \\ x &\mapsto M = (m_{ij}) \text{ mit } m_{ij} = x_{(j-1)\cdot n+i}. \end{aligned}$$

Anschaulich bildet man mit dieser Abbildung einen Vektor der Länge n^2 auf eine Matrix M ab, indem jeweils n Elemente eine Spalte der Matrix bilden. Betrachtet man $y = \mathcal{K}^{-1}(M^G)$ der kanonischen Repräsentation M^G des Graphen G, so ist y ein Vektor der Länge n^2 mit e Einsen und $n^2 - e$ Nullen. Seien $p_1, \ldots p_e$ die Positionen der Einsen in y, das heißt

$$p_i = j \text{ mit } y_j = 1 \text{ und } \sum_{l=1}^{j} y_l = i.$$

Mit Hilfe der Positionen $p_1, \ldots p_e$ definiert man die Projektionsabbildung $\mathcal{P}_G : \mathbb{R}^{n^2} \to \mathbb{R}^e$ durch $\mathcal{P}_G(y) = x$ mit $x_i = y_{p_i}$. Als Verknüpfung der Abbildungen $\mathcal{K}$ und $\mathcal{P}_G$ ergibt sich die Abbildung

$$\begin{aligned} \mathcal{S}_G : \mathbb{R}^e &\to R_G \\ x &\mapsto \mathcal{K}(\mathcal{P}_G^{-1}(x)) \end{aligned}$$

mit $R_G = \{K = (k_{ij}) \in M(n, n, \mathbb{R}) \mid K \text{ repräsentiert } G\}$. Die Abbildung $\mathcal{S}_G$ erzeugt also Matrizen, die den Graphen G repräsentieren, wobei die Stellen der Matrix, die nicht durch eine Null vorgegeben sind, durch Elemente des Vektors x besetzt werden. Die so erzeugte Matrix $S = (s_{ij}) \in M(n, n, \mathbb{R})$ ist eine obere Dreiecksmatrix mit $s_{ii} = 0$ für alle i, denn es gilt $(i, i) \notin E_G$ für alle i.

Die Matrix $S \in \mathcal{S}_G(\mathbb{R}^e)$ ist eine obere Dreiecksmatrix mit Diagonale Null, also ist diese Matrix insbesondere eine Matrix ohne vollen Rang. Es werden nun die Diagonalelemente der erzeugten Matrizen S besetzt, wobei sicher gestellt wird, dass die Diagonalelemente der Matrix $Q := S^t \cdot S$ die Bedingung 3 erfüllen. Hierzu bedarf es noch einiger Definitionen. Seien $\tilde{R}_G \subset R_G$ und $\tilde{\mathbb{R}}^e \subset \mathbb{R}^e$ definiert durch

$$\begin{aligned} \tilde{R}_G &:= \{K = (k_{ij}) \in M(n, n, \mathbb{R}) \mid K \in R_G \text{ und } \sum_{l=1}^{n} k_{lj}^2 < 1 \text{ für alle } j\} \\ \tilde{\mathbb{R}}^e &:= \{x \in \mathbb{R}^e \mid \mathcal{S}_G(x) \in \tilde{R}_G\} \end{aligned}$$

Dann definiert man die Funktion

$$\begin{aligned} \mathcal{D}_G : \tilde{\mathbb{R}}^e \times \{-1, 1\}^n &\to M(n, n, \mathbb{R}) \\ (x, y) &\mapsto D = (d_{ij}) \end{aligned}$$

mit $d_{ij} = 0$ für alle $i \neq j$ und

$$d_{ii} = y_i \cdot \sqrt{1 - \sum_{l=1}^{n} s_{li}^2}.$$

Hierbei ist $S = (s_{ij}) \in M(n, n, \mathbb{R})$ definiert ist als $S = \mathcal{S}_G(x)$. Die Abbildung $\mathcal{D}_G$ ist wegen der Eigenschaften der Menge $\tilde{\mathbb{R}}^e$ wohldefiniert. Verknüpft man die Abbildungen $\mathcal{D}_G$ und $\mathcal{S}_G$, so erhält man schließlich

$$\begin{aligned} \mathcal{W}_G : \tilde{\mathbb{R}}^e \times \{-1, 1\}^n &\to OD(n) \\ (x, y) &\mapsto \mathcal{D}_G(x, y) - \mathcal{S}_G(x) \end{aligned}$$

Nach diesen Vorbereitungen kann die zu optimierende Abbildung definiert werden:

$$\begin{array}{rcl} OP_A : \tilde{\mathbb{R}}^e \times \{-1,1\}^n & \rightarrow & \mathbb{R} \\ (x,y) & \mapsto & \|A - (\mathcal{W}_G(x,y))^t \cdot \mathcal{W}_G(x,y)\| \end{array}$$

Der DAG G ist hierbei eine Prämoralisierung des die Matrix A repräsentierenden Graphen H. Als Norm wird, wie oben beschrieben, die Frobeniusnorm benutzt.

Wenn man die Funktion OP_A minimieren möchte, versucht man, die Einträge einer oberen Dreiecksmatrix K mit vollem Rang so zu besetzen, dass der Abstand zwischen einer Matrix A und der Matrix $Q = K^t \cdot K$ bezüglich der Frobeniusnorm minimiert wird. Die Matrix K repräsentiert hierbei (mit Ausnahme der Diagonalelemente) eine Prämoralisierung des Graphen H, den die Matrix A repräsentiert. Durch die Restriktion von x und die Wahl der Diagonalelemente der Matrix K wird sicher gestellt, dass die Matrix Q die Eigenschaft 3 erfüllt, die Diagonalelemente also 1 sind. Dies wird im folgenden Lemma gezeigt.

Lemma 4.4.1 *Für $x \in \tilde{\mathbb{R}}^e$ sei $K := \mathcal{W}_G(x)$. Sei $Q = (q_{ij}) \in M(n,n,\mathbb{R})$ definiert als $Q = K^t \cdot K$. Dann gilt $q_{ii} = 1$ für alle i.*

Beweis: Der Beweis ist eine direkte Folgerung aus der Definition von $\tilde{\mathbb{R}}^e$. Sei also $x \in \tilde{\mathbb{R}}^e$, und $K = (k_{ij}) \in M(n,n,\mathbb{R})$ sei definiert als $K := \mathcal{W}_G(x)$. Nach Definition von $\mathcal{W}_G$ lässt sich K schreiben als $K = D - S$ mit $D = (d_{ij}) \in M(n,n,\mathbb{R})$ definiert als $D = \mathcal{D}_G(x)$ und $S = (s_{ij}) \in M(n,n,\mathbb{R})$ definiert als $\mathcal{S}_G(x)$. Es gilt $s_{ii} = 0$ für alle i, weil G azyklisch ist. Somit gilt $k_{ii} = d_{ii}$ für alle i und $k_{ij} = -s_{ij}$ für alle $i \neq j$. Daraus folgt zusammengenommen für alle i

$$\begin{array}{rcl} q_{ii} & = & \sum_{l=1}^{n} (k_{li})^2 \\ & = & (d_{ii})^2 + \sum_{l=i+1}^{n} (k_{li})^2 \\ & = & 1 - \sum_{l=i+1}^{n} s_{li}^2 + \sum_{l=i+1}^{n} (k_{li})^2 \\ & = & 1 \end{array}$$

q.e.d.

4.4.1 Validierung der Optimierungsverfahren

Zum Optimieren der Funktion OP_A werden die in Kapitel 2 vorgestellten Verfahren BFGS und Nelder-Mead benutzt. Um zu testen, welches Verfahren für OP_A ein besseres Ergebnis liefert, werden vier verschiedene Ansätze verwendet. Alle Graphen, die als Grundlage für die Simulationen dienen, sind prämoralisierbare Graphen $H = (V, E)$, die aus realen Microarray-Daten geschätzt worden sind (siehe Kapitel 5.2). Unter diesen Graphen wurden für diese Studie die Graphen H gewählt, die mindesten 20 Kanten haben und für die ein Knoten v existert mit $\deg_H(v) > 1$.

- **Ansatz 1:** Für jeden Graphen H wird eine zufällige, den Graphen repräsentierende Matrix A erzeugt, wobei für die zu den Kanten korrespondierenden Einträge der Matrix Werte zwischen 0.8 und 1 oder -1 und -0.8 gewählt werden. Für diese Matrix A wird die Funktion OP_A optimiert.

- **Ansatz 2:** Als einzige Änderung zu Ansatz 1 wird nicht der gesamte Graph benutzt, um die Matrix A zu erstellen, sondern nur die größte Zusammenhangskomponente. Dies ist sinnvoll, da man später auch so vorgehen möchte, dass man jede Zusammenhangskomponente einzeln optimiert.

- **Ansatz 3:** In diesem Ansatz wird ebenfalls nur die größte Zusammenhangskomponente des Graphen benutzt. Allerdings wird in diesem Fall die Matrix A nicht vollkommen zufällig besetzt, sondern so, dass sie selbst schon positiv definit ist. Dies wird erreicht, indem eine wie oben erstellte Matrix A durch Addition auf den Diagonalelementen diagonaldominant gemacht und dann zu einer Korrelationsmatrix normiert wird.

- **Ansatz 4:** In diesem Ansatz wird ebenfalls eine positiv definite Matrix A vorgegeben. In diesem Fall ist die Matrix A schon ein Resultat aus einer einmaligen Optimierung durch den Algorithmus. Dadurch ist sicher gestellt, dass es für die Matrix eine Zerlegung gibt, bei der die obere Dreiecksmatrix eine Prämoralisierung von H repräsentiert.

Somit sind die Matrizen aus Ansatz 1 und Ansatz 2 nicht notwendigerweise positiv definit, und die Matrizen aus Ansatz 3 und Ansatz 4 sind positiv definit. Die Ergebnisse sind in Abbildung 4.6 dargestellt. Man erkennt hier eindeutig, dass in allen Situationen das BFGS-Verfahren bessere Ergebnisse liefert als das Nelder-Mead-Verfahren. Das BFGS-Verfahren erzielt sehr gute Ergebnisse bei den Ansätzen 3 und 4 und gute Ergebnisse bei den Ansätzen 1 und 2. Besonders das sehr gute Ergebniss in Setting 3 legt nahe, dass man eine positiv definite Matrix immer durch eine Matrix der Form $Q = K \cdot K^t$ annähern kann, wobei K eine beliebige Prämoralisierung G repräsentiert. Auf Grund der Studien wird in den folgenden Berechnungen das BFGS-Verfahren benutzt.

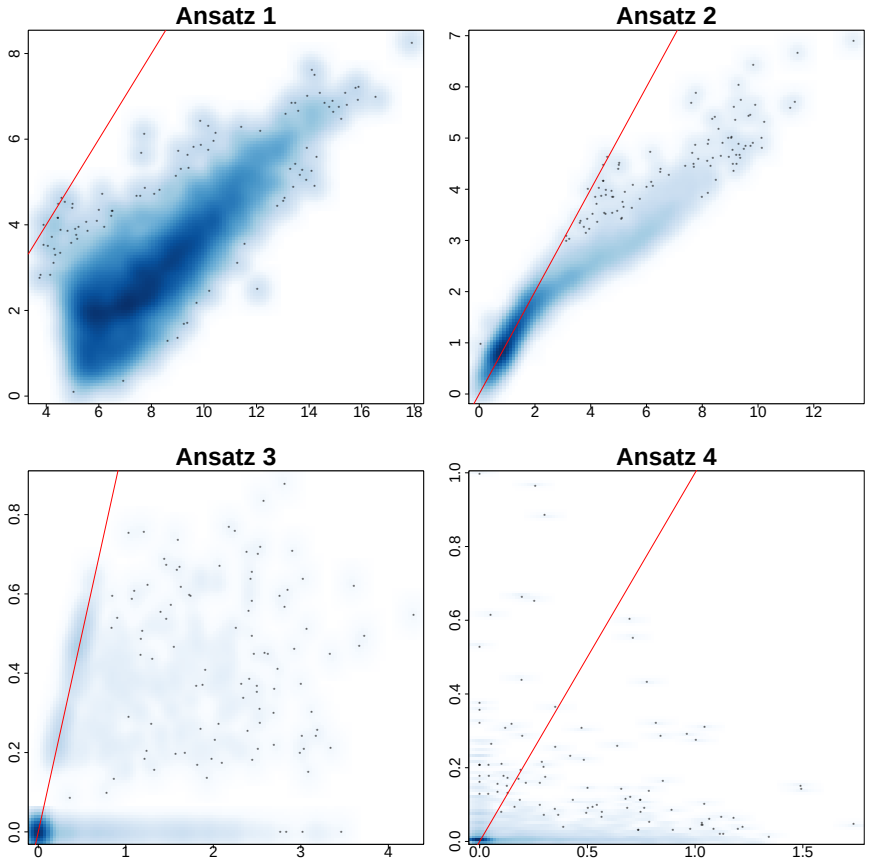

Abbildung 4.6: Ergebnis des Simulationsansatzes zwischen Nelder-Mead und BFGS für die Frobeniusnorm. Auf der x-Achse ist das Ergebnis mit dem Nelder-Mead-Verfahren gezeigt und auf der y-Achse die Optimierung mit dem BFGS-Verfahren.

4.4.2 Einbindung von partiellen Varianzen

Im letzten Abschnitt wurde für einen Graphen H eine den Graphen repräsentierende symmetrische Matrix $A = (a_{ij}) \in M(n, n, \mathbb{R})$ vorgegeben, mit $a_{ii} = 1$ für alle i und $\mid a_{ij} \mid \leq 1$ für alle i und j. Diese Matrix wurde interpretiert als eine Vorgabe der partiellen Korrelationen zwischen den Variablen. Es wurde dann die Matrix $\tilde{A} = (\tilde{a}_{ij}) \in M(n, n, \mathbb{R})$ definiert als $\tilde{a}_{ij} := -a_{ij}$ für alle $i \neq j$ und $\tilde{a}_{ii} := a_{ii} = 1$ für alle i, und es wurde eine den Graphen H repräsentierende positiv definite Matrix $K^t \cdot K = \Omega = (\omega_{ij}) \in M(n, n, \mathbb{R})$ erstellt, mit $\omega_{ii} = 1$ für alle i und $\mid \omega_{ij} \mid \leq 1$ für alle i, j. Ω beziehungsweise K wurde so optimiert, dass die Frobeniusnorm zwischen Ω und $\tilde{A}$ minimiert wird.

In diesem Ansatz sind partielle Varianzen, die Diagonalelemente der Inversen einer Kovarianzmatrix, noch nicht betrachtet worden. Das bedeutet, wenn man mit Hilfe der Matrix K aus der Normalverteilung $N(0, \Omega^{-1})$ Daten erzeugt, so werden die partiellen Varianzen 1 sein. Es ist aber problemlos möglich, den Ansatz so zu erweitern, dass man

auch partielle Varianzen vorgeben kann.

Angenommen, für jedes i ist eine partielle Varianz $1/v_i > 0$ vorgegeben. Man definiert eine Diagonalmatrix $D = (d_{ij}) \in M(n, n, \mathbb{R})$ durch $d_{ii} = \sqrt{v_i}$ für alle i und die Matrix $Q = (q_{ij}) \in M(n, n, \mathbb{R})$ als

$$Q = D \cdot \Omega \cdot D.$$

Diese Matrix ist positiv definit und sie repräsentiert den Graphen H. Zudem gilt

$$q_{ii} = \sum_{l=1}^{n} d_{il} \cdot (\sum_{k=1}^{n} \omega_{lk} \cdot d_{ki}) = d_{ii} \cdot (\sum_{k=1}^{n} \omega_{ik} \cdot d_{ki}) = d_{ii} \cdot \omega_{ii} \cdot d_{ii} = d_{ii}^2 = v_i.$$

Die Matrix Q kann interpretiert werden als Präzessionsmatrix, die zu einem Gaußschen Graphischen Modell induziert durch den Graphen H gehört. Nach Definition sind die partiellen Varianzen für dieses Modell gegeben durch $1/q_{ii} = 1/v_i$, entsprechen also den vorgegebenen partiellen Varianzen. Die Korrelationsmatrix $f(Q)$ von Q ist Ω, denn wie gezeigt lässt sich die Korrelationsmatrix $f(Q)$ darstellen als $f(Q) = T \cdot Q \cdot T$ mit T Diagonalmatrix und $t_{ii} = \frac{1}{\sqrt{q_{ii}}} = \frac{1}{\sqrt{v_i}}$. Es gilt somit:

$$f(Q) = T \cdot Q \cdot T = (T \cdot D) \cdot \Omega \cdot (D \cdot T) = \Omega.$$

Für den nachfolgenden Algorithmus zur Erzeugung von Daten aus einer multivariaten Normalverteilung $N(\mu, \Sigma)$ ist es vorteilhaft, wenn die Präzessionsmatrix Σ^{-1} in einer Zerlegung der Form $K^t \cdot K$ vorliegt, wobei K eine obere Dreiecksmatrix mit vollem Rang ist. Da aber nach Konstruktion für Ω eine solche Zerlegung mit der Matrix K schon vorliegt, gilt

$$Q = D \cdot \Omega \cdot D = D^t \cdot K^t \cdot K \cdot D = (KD)^t \cdot (KD).$$

Die Matrix KD ist eine obere Dreiecksmatrix mit vollem Rang, da $v_i > 0$ für alle i, also hat man nun auch eine gewünschte Zerlegung für Q.

4.4.3 Algorithmus zum Erzeugen von multivariat normalverteilten Daten

In Abschnitt 4.2 wurde der PddP-Algorithmus vorgestellt, mit dem man positiv definite Matrizen mit Nebenbedingungen erzeugen kann. Zusätzlich wurde gezeigt, wie man die freien Elemente besetzen kann, um eine positiv definite Matrix zu erzeugen, die bezüglich der Frobeniusnorm einen geringen Abstand zu einer vorgegebenen Matrix hat. Das Ziel eines Simulationsansatzes liegt aber nicht nur in der Erzeugung einer Matrix, sondern man möchte auch aus der zugehörigen Verteilung Daten erzeugen (siehe Abbildung 4.5). Dies ist mit Hilfe des PddP-Algorithmus aber sehr einfach, da in diesem Fall eine Zerlegung der Matrix vorliegt und man so ohne weitere Berechnungen das in Kapitel 2.2. eingeführte Simulationsverfahren nutzen kann.

Im Detail sei ein ungerichteter prämoralisierbarer Graph $H = (V, E_H)$ mit $|V| = n$ gegeben. Sei $A = (a_{ij}) \in M(n, n, \mathbb{R})$ eine nicht notwendigerweise positiv definite Matrix, die H repräsentiert, also insbesondere symmetrisch ist. Zudem sei $a_{ii} = 1$ für alle i und

$|a_{ij}| \leq 1$ für alle i und j. Zusätzlich seien $v_i > 0$ für $i = 1, \ldots n$ vorgegebene Werte. Man möchte nun Daten x aus einer multivariaten Normalverteilung $N(0, \Sigma)$ erzeugen, wobei die partiellen Varianzen dieser Verteilung durch $1/v_i$ gegeben sein sollen und die partielle Korrelationsstruktur durch A. Das Schema hierfür ist nun wie folgt:

Algorithmus A

1. Optimiere die Funktion OP_A mit Hilfe des BFGS-Verfahrens. Notiere die erzeugten Matrizen Ω und K. Insbesondere wird hierfür eine Prämoralisierung des Graphen H benötigt.

2. Definiere die Diagonalmatrix $D = (d_{ij}) \in M(n, n, \mathbb{R})$ mit $d_{ii} = \sqrt{v_i}$ für alle $i = 1, \ldots, n$.

3. Erzeuge einen zufälligen Vektor z der Länge n mit unabhänging standard-normalverteilten Werten.

4. Löse das Gleichungssystem $(KD) \cdot x = z$. Die Lösung x dieses Gleichungssystems ist multivariat normalverteilt mit Mittelwert 0 und Kovarianzmatrix $(D \cdot \Omega \cdot D)^{-1}$.

Dieser Algorithmus wird im nachfolgenden Kapitel mehrfach Anwendung finden, wenn es nötig ist, normalverteilte Daten zu erzeugen. Der Schwerpunkt wird hier auf der Validierung von Algorithmen liegen, die Netzwerke aus Microarray-Daten erzeugen. Aber auch für andere Situation ist das Erzeugen von multivariat normalverteilten Daten mit Nebenbedingungen sehr nützlich. Da der beschriebene Algorithmus aber nur für prämoralisierbare Graphen funktioniert, und nicht jeder Graph diese Eigenschaft besitzt (siehe Kapitel 3), muss auch getestet werden, wie groß der Anteil der prämoralisierbaren Graphen ist, zumindest in der Menge der graphischen Strukturen, die aus Microarray-Daten geschätzt werden, denn Algorithmus A soll vor allem für diese Strukturen angewandt werden. Auch diese Untersuchung ist Teil von Kapitel 5.

Kapitel 5

Relevanz prämoralisierbarer Graphen für biologische Phänomene

In diesem Kapitel sollen Strukturen untersucht werden, die bei der Schätzung von Netzwerken aus Microarray-Daten auftreten. Der Fokus wird hier auf den in Kapitel 2 beschriebenen Algorithmus von Schäfer und Strimmer[64] gelegt, aber alle Untersuchungen können natürlich auch mit anderen Algorithmen durchgeführt werden. In den meisten untersuchten Fragestellungen finden Simulationen von Daten aus multivariaten Normalverteilungen Anwendung, so dass dieses Kapitel als eine Anwendung der im letzten Kapitel eingeführten Algorithmen gesehen werden kann.

5.1 Netzwerke auf Spotebene und Netzwerke auf Genebene

Bevor man mit der Schätzung eines Netzwerkes aus Genexpressionsdaten startet, muss man entscheiden, was die Knoten des Netzwerkes repräsentieren sollen. Hierbei gibt es zwei naheliegende Möglichkeiten: Entweder jeder Knoten repräsentiert einen Spot des Chips, oder aber jeder Knoten repräsentiert ein Gen. In diesem Zusammenhang versteht man hier unter einem Gen eine Entrez ID, denn eine Entrez ID ist mit einem Gen assoziert und unter einer Entrez ID sind verschiedene Sequenzen zusammengefasst, von denen einzelne durch Spots auf Microarray-Chips repräsentiert sind.

In einem Microarray-Experiment hat der genutzte Chip in der Regel mehrere Spots, die auf eine Entrez ID abgebildet werden. Dies bedeutet, man hat eine nicht injektive und nicht surjektive Abbildung von der Menge der Spots eines Chips auf die Menge der Entrez IDs beziehungsweise die Menge der Gene.

Unter der Voraussetzung, dass die ein Gen repräsentierenden Spots die gleiche Expression messen, verschiedene Spots eines Chips also nicht verschiedene Splice-Varianten des gleichen Gens abdecken, kann man davon ausgehen, dass diese Spots eine hohe Korrelation haben, da es sich dann bei den Unterschieden zwischen den Spots vor allem um eine technische Varianz handelt, die in der Regel geringer ist als eine biologische Varianz. Möchte

man also ein Netzwerk auf Spotebene schätzen, so gehen in die Berechnungen mehrere Variablen ein, die eine starke Korrelation haben. Dies kann zu Problemen bezüglich der partiellen Korrelation führen, wie im Folgenden gezeigt wird (siehe auch Kapitel 1, Abbildung 1.3). Es wird nun eine Simulationsstudie durchgeführt, die zeigt, dass Schätzungen auf Genebene zu besseren Ergebnissen führen als Schätzungen auf Spotebene.

Zur Motivation wird nun ein erstes Beispiel angegeben, in dem A und B Messwerte von zwei Spots sind, die das gleiche Gen repräsentieren. Die Expression dieses virtuellen Gens ändert sich mit der Zeit linear. C und D sind die Messwerte von zwei Spots, die ein zweites Gen repräsentieren. Dieses verhält sich gegenläufig zu Gen 1.

$$\begin{aligned} Gen_1 &= 10 - Gen_2 \\ A, B &\sim N(Gen_1, 0.1) \\ C, D &\sim N(Gen_2, 0.1) \end{aligned}$$

Man kann nun die partiellen Korralationen auf Spotebene oder aber auf Genebene berechnen. Im ersten Fall benutzt man die Expressionswerte aller Spots. Dann wird der empirische Schätzer für die Kovarianzmatrix benutzt und mit diesem durch Invertierung die partielle Korrelationsmatrix berechnet. Es ergibt sich in diesem Beispiel für eine große Menge an erzeugten Daten aus der obigen Verteilung:

```
      A     B     C     D
A  1.00  0.42 -0.08 -0.47
B  0.42  1.00 -0.46 -0.12
C -0.08 -0.46  1.00  0.45
D -0.47 -0.12  0.45  1.00
```

Möchte man die partielle Korrelation auf Genebene schätzen, so wird der Einfachheit halber pro Gen ein repräsentativer Spot gewählt. Andere Vorgehensweisen wie die Verwendung der Spotexpressionswerte zur Erstellung eines Genexpressionswertes mit Hilfe eines linearen Modelles oder die einfache Bildung des Mittelwertes aller Spots sind denkbar und werden im zweiten Simulationsbeispiel teilweise angewandt. In diesem Beispiel werden die Spots A und C als Repräsentanten gewählt. Dann wird die empirische Kovarianzmatrix berechnet und aus dieser durch Invertierung die partiellen Korrelationen abgeleitet. Es ergibt sich in diesem Fall:

```
   A  C
A  1 -1
C -1  1
```

Bei dem Resultat auf Genebene ist die negative partielle Korrelation zwischen den Genen sehr deutlich erkennbar, während bei dem Resultat auf Spotebene dieses nicht der Fall ist. Dies gibt einen ersten Hinweis, dass es sinnvoller ist, die partiellen Korrelationen auf Genebene und nicht auf Spotebene zu schätzen.

Der Unterschied zwischen Schätzungen auf Genebene und Spotebene wird nun in einem großen Simulationsansatz weiter untersucht. Zu Grunde liegen hier Graphen, die aus realen Microarray-Datensätzen mit Hilfe des Algorithmus von Schäfer und Strimmer geschätzt worden sind (siehe auch Kapitel 5.2). Unter allen geschätzten Graphen werden die Graphen $H = (V_H, E_H)$ ausgewählt, die folgende Eigenschaften erfüllen:

- H wurde bei einer q-Wert-Schranke von 0.1 oder 0.2 erzeugt (siehe Kapitel 5.2)
- H ist prämoralisierbar
- $|E_H| > 1$
- Es gibt einen Knoten $v \in V_H$ mit $N_H(v) > 1$.

Auf Grund dieser Reduktion ergeben sich 3710 Graphen. Die Strukturen der zu Grunde liegenden Graphen werden allgemein in Abschnitt 2 dieses Kapitels genauer untersucht. Für jeden dieser Graphen $H = (V_H, E_H)$ mit $V_H = \{g_1, \ldots, g_n\}$ werden 5 verschiedene Präzessionsmatrizen $\Omega^H = (\omega_{ij}^H) \in M(n, n, \mathbb{R})$ erzeugt, die durch H gegebene Nebenbedingungen besitzen. Dies geschieht mit dem PddP-Algorithmus (siehe Kapitel 4), wobei die vorgegebenen Matrizen A hier zufällige Werte zwischen -1 und -0.1 oder 0.1 und 1 besitzen. Die vorgegebenen partiellen Varianzen werden zufällig nahe bei 1 erzeugt. Mit Hilfe der Matrix Ω^H generiert man eine multivariate Normalverteilung $N_H(\mu, \Sigma^H)$, mit

$$\Sigma^H = (\Omega^H)^{-1} \text{ und } \mu \sim N(0, 5).$$

Aus dieser Normalverteilung werden 1000 Datenpunkte mit Hilfe von Algorithmus A erzeugt. Es ergibt sich die Matrix $R^H = (r_{ij}^H) \in M(n, 1000, \mathbb{R})$. Diese Matrix repräsentiert die Genexpression der n Gene in 1000 verschiedenen Datenpunkten. Die Anzahl wurde hier so groß gewählt, um den empirischen Schätzer für die Kovarianzmatrix nutzen zu können.

Man bezeichnet mit $g_1^H, \ldots, g_n^H$ die verschiedenen Zeilen aus R^H, also die verschiedenen Gene. Um die Expression eines Chips zu erzeugen, auf dem einige der Gene $g_1^H, \ldots, g_n^H$ durch mehrere Spots vertreten sind, wird zuerst für jedes Gen die Anzahl der Spots generiert, durch die es auf dem Chip repräsentiert ist. Diese sei poissonverteilt, das heißt für

$$X(g_i^H) := \text{ Menge der repräsentierenden Spots für Gen } g_i^H$$

gilt

$$X(g_i^H) \sim Po(0.5) + 1.$$

Durch den zusätzlichen Summanden wird sicher gestellt, dass immer mindestens 1 Spot pro Gen g_i auf dem Chip vertreten ist. Für jedes Gen g_i^H werden dann die Expressionen der Spots simuliert. Dazu erzeugt man für jedes g_i^H genau $X(g_i^H)$ Datenpunkte einer multivariaten Normalverteilung mit Mittelwert $R^H_{g_i^H,-}$ und Varianz 1. Hieraus ergibt sich

eine Matrix $\tilde{R}^H = (\tilde{r}_{ij}^H) \in M(\tilde{n}, 1000, \mathbb{R})$ mit $\tilde{n} := \sum_i X(g_i^H)$. Mit Hilfe dieser Matrix werden nun drei verschiedene Ansätze verglichen, mit denen eine partielle Korrelationsmatrix geschätzt werden kann:

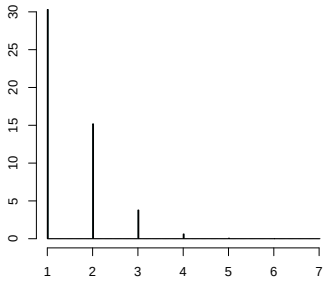

Abbildung 5.1: Verteilung von zufällig erzeugten Werten von X

1. **Jeden Spot benutzen:** Aus der kompletten Matrix $\tilde{R}^H$ wird die empirische Kovarianzmatrix geschätzt. Diese wird invertiert und die sich daraus ergebende Matrix wird zu einer Korrelationsmatrix $K^{\tilde{R}^H,1} = (k_{ij}^{\tilde{R}^H,1}) \in M(\tilde{n}, \tilde{n}, \mathbb{R})$ normiert. Aus dieser Matrix wird die Matrix $\hat{K}^{\tilde{R}^H,1} = (\hat{k}_{ij}^{\tilde{R}^H,1}) \in M(n, n, \mathbb{R})$ erstellt, wobei $\hat{k}_{ij}^{\tilde{R}^H,1}$ definiert ist als Mittelwert aus den Werten der Matrix $K_{m(i),m(j)}^{\tilde{R}^H,1}$ mit $m(i) :=$ Index der das Gen i repräsentierenden Zeilen in $K^{\tilde{R}^H,1}$.

2. **Einen Spot benutzen:** Für jedes Gen wird ein Spot ausgewählt, das bedeutet die Matrix $\tilde{R}^H$ wird auf eine Zeile pro Gen reduziert. Für die Spotauswahl wird pro Beobachtung der Median aller repräsentierenden Spots berechnet. Es wird dann der Spot gewählt, der bezüglich der Frobeniusnorm den geringsten Abstand zum Median aufweist. Gilt dies für mehrere Spots, so wird ein Spot zufällig gewählt. Dies ist insbesondere der Fall, wenn es zwei ein Gen repräsentierende Spots gibt. Für die resultierende Matrix wird dann die empirische Kovarianzmatrix geschätzt. Diese wird invertiert und die sich daraus ergebende Matrix wird zu einer Korrelationsmatrix $K^{\tilde{R}^H,2} = (k_{ij}^{\tilde{R}^H,2}) \in M(n, n, \mathbb{R})$ normiert.

3. **Mittelwerte der Spots benutzen:** Für jedes Gen wird aus allen dieses Gen repräsentierenden Zeilen der Matrix $\tilde{R}^H$ der Mittelwert pro Beobachtung berechnet. Für die resultierende Matrix wird die empirische Kovarianzmatrix geschätzt. Diese wird invertiert und die sich daraus ergebende Matrix wird zu einer Korrelationsmatrix $K^{\tilde{R}^H,3} = (k_{ij}^{\tilde{R}^H,3}) \in M(n, n, \mathbb{R})$ normiert.

Für jeden der drei Ansätze werden die Resultate der Schätzungen mit der ursprünglichen Matrix Ω^H verglichen. Hierzu betrachtet man den Abstand bezüglich der Frobeniusnorm zwischen Ω^H und $\hat{K}^{\tilde{R}^H,1}$,$K^{\tilde{R}^H,2}$ oder $K^{\tilde{R}^H,3}$. In Abbildung 5.2 sind jeweils zwei der

Abstände für alle erzeugten Matrizen $\tilde{R}^H$ (mit zu Grunde liegender Matrix Ω^H) gegeneinander aufgetragen. Was sich in dem ersten Beispiel angedeutet hat, wird nun bestätigt. Man erkennt, dass sowohl die Spotauswahl als auch die Mittelwertbildung bessere Resultate liefern als die Nutzung aller Spots. Zwischen der Mittelwertbildung und der Spotauswahl gibt es keine großen Unterschiede.

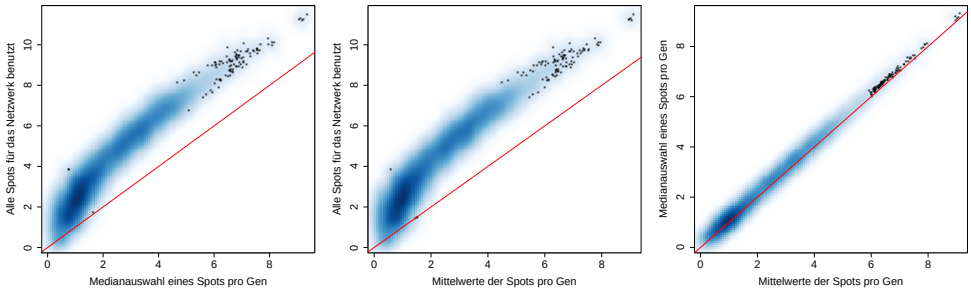

Abbildung 5.2: Für alle erzeugten Matrizen $\tilde{R}^H$ (mit zu Grunde liegender Matrix Ω^H) sind jeweils zwei der Abstände (Frobeniusnorm) zwischen Ω^H und $\hat{K}^{\tilde{R}^H,1}$,$K^{\tilde{R}^H,2}$ oder $K^{\tilde{R}^H,3}$ gegeneinander aufgetragen.

Man möchte die Unterschiede nun speziell für die Werte betrachten, bei der für die ursprüngliche Matrix Ω^H ein Wert ungleich Null vorhanden ist. Hierzu betrachtet man alle i und j mit $\{g_i, g_j\} \in E_H$ und berechnet

$$\begin{aligned} M_{i,j,\tilde{R}^H,1} &:= \mid \omega_{ij}^H - \hat{k}_{ij}^{\tilde{R}^H,1} \mid \\ M_{i,j,\tilde{R}^H,2} &:= \mid \omega_{ij}^H - k_{ij}^{\tilde{R}^H,2} \mid \\ M_{i,j,\tilde{R}^H,3} &:= \mid \omega_{ij}^H - k_{ij}^{\tilde{R}^H,3} \mid \end{aligned}$$

Die Resultate dieser Vergleiche sind in den Abbildungen 5.3, 5.4 und 5.5 dargestellt. Hier sind jeweis zwei der Werte $M_{\cdot,\cdot,\cdot,1}$, $M_{\cdot,\cdot,\cdot,2}$ und $M_{\cdot,\cdot,\cdot,3}$ gegeneinander aufgetragen, wobei unterschieden wird, ob für die Werte $M_{i,j,\tilde{R}^H,\cdot}$ gilt, dass die Gene g_i und g_j in $\tilde{R}^H$ nur durch einen Spot repräsentiert worden sind oder nicht. Es zeigt sich, dass die drei vorgestellten Ansätze keine Unterschiede aufweisen, wenn beide Gene nur durch einen Spot repräsentiert werden. Das ist zu erwarten. Wenn aber wenigstens eines der Gene durch zwei oder mehrere Spots repräsentiert wird, so sieht man auch hier die Vorteile bei Ansatz 2 und 3. Zudem erkennt man einen leichten Vorteil, wenn man Mittelwerte und keine repräsentierenden Spots benutzt. Insgesamt wird mit diesen Ergebnissen gezeigt, dass es sinnvoller ist, die Netze auf Genebene zu schätzen. Dies ist auch die Vorgehensweise in den nächsten Abschnitten: wenn im Folgenden davon gesprochen wird, dass ein Netzwerk aus Microarray-Daten erzeugt wird, so wird dies immer auf der Ebene der Entrez IDs geschehen.

Die vorliegende Simulationsstudie ist ein erster Schritt, um Unterschiede zwischen der Erzeugung eines Netzwerkes auf Genebene und auf Spotebene zu untersuchen. Natürlich

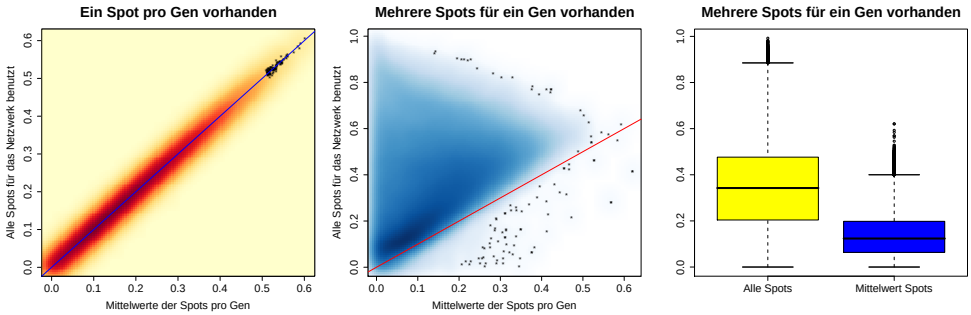

Abbildung 5.3: Vergleich Mittelwert der Spots für ein Gen gegen alle Spots für ein Gen. Aufgetragen ist $M_{i,j,\tilde{R}^H,3}$ gegen $M_{i,j,\tilde{R}^H,1}$ für jede gegebene Matrix $\tilde{R}^H$ mit zu Grunde liegender Matrix Ω^H. Es wird unterschieden zwischen Genpaaren g_i und g_j, bei denen beide Gene durch nur einen Spot in $\tilde{R}^H$ repräsentiert werden, und allen übrigen Genpaaren.

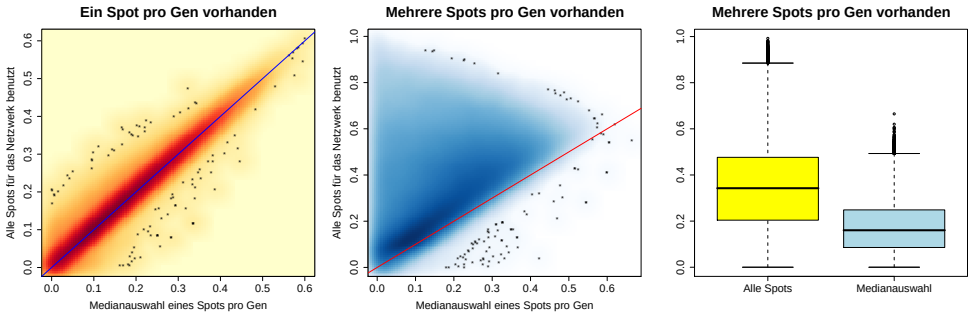

Abbildung 5.4: Vergleich Auswahl eines Spots für ein Gen gegen alle Spots für ein Gen. Aufgetragen ist $M_{i,j,\tilde{R}^H,2}$ gegen $M_{i,j,\tilde{R}^H,1}$ für jede gegebene Matrix $\tilde{R}^H$ mit zu Grunde liegender Matrix Ω^H. Es wird unterschieden zwischen Genpaaren g_i und g_j, bei denen beide Gene durch nur einen Spot in $\tilde{R}^H$ repräsentiert werden, und allen übrigen Genpaaren.

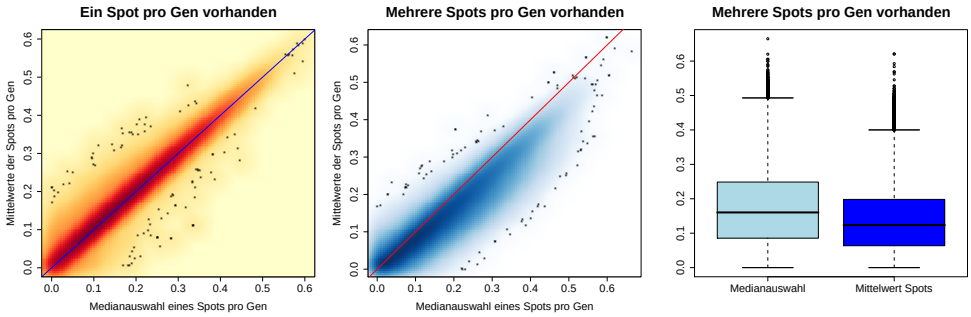

Abbildung 5.5: Vergleich Auswahl eines Spots für ein Gen gegen Mittelwert aller Spots für ein Gen. Aufgetragen ist $M_{i,j,\tilde{R}^H,2}$ gegen $M_{i,j,\tilde{R}^H,3}$ für jede gegebene Matrix $\tilde{R}^H$ mit zu Grunde liegender Matrix Ω^H. Es wird unterschieden zwischen Genpaaren g_i und g_j, bei denen beide Gene durch nur einen Spot in $\tilde{R}^H$ repräsentiert werden, und allen übrigen Genpaaren.

gibt es viele Möglichkeiten, die Untersuchung auszuweiten. Beispielsweise sind andere Modelle für die Anzahl der auf einem Chip vorliegenden Spots für ein Gen denkbar, man kann auch von realen Anzahlen ausgehen, wie sie auf einem existierenden Oligo- oder cDNA-Chip vorliegen. Auch bei der Wahl der übrigen zu Grunde liegenden Parameter gibt es Spielraum. All diese Möglichkeiten zu testen liegt aber nicht im Fokus dieser Studie, die mehr als Anregung gesehen werden sollte, sich näher mit der Thematik zu befassen. In dieser Arbeit soll vor allem gezeigt werden, dass es Unterschiede zwischen den auf den verschiedenen Ebenen geschätzten Netzwerken gibt. Zudem musste für die Untersuchungen in den folgenden Abschnitten eine Entscheidung getroffen werden, ob man Netzwerke auf der Spot- oder Genebene erzeugen soll.

5.2 Prämoralisierbare Graphen in Microarray-Daten

In Kapitel 4 wurde der PddP-Algorithmus vorgestellt, der es erlaubt, eine zufällige positiv definite Matrix mit Nebenbedingungen zu erzeugen, falls der Graph, der die Nebenbedingungen induziert, prämoralisierbar ist. Es wurde in Kapitel 3 gezeigt, dass nicht jeder Graph prämoralisierbar ist und es wurde ein Algorithmus vorgestellt, der es ermöglicht zu entscheiden, ob ein Graph prämoralisierbar ist oder nicht. Eine Hauptmotivation für den PddP-Algorithmus liegt darin, eine Matrix $\Omega \in SPR(n, H)$ zu erzeugen, wobei der Graph H eine mögliche partielle Korrelationsstruktur zwischen Genen repräsentieren soll. In Kapitel 5.3 wird auf die Wahl des Graphen detailliert eingegangen. Da also der PddP-Algorithmus vor allem für Graphen H genutzt werden soll, die partielle Korrelationsstrukturen zwischen Genen repräsentieren, der Algorithmus aber voraussetzt, dass der Graph H prämoralisierbar ist, ist es wichtig zu wissen, wie groß die Menge der prämoralisierbaren Graphen ist, die partielle Korrelationsstrukturen repräsentieren. Wie solche Strukturen

zwischen Genen aussehen, ist größtenteils noch nicht bekannt. Es gibt zwar Kenntnisse, dass gewisse Gene mit anderen Genen interagieren, aber häufig ist nicht einmal die Frage beantwortet, ob es sich dabei um direkte oder indirekte Interaktionen handelt. Also muss hier eine empirische Untersuchung durchgeführt werden. Man möchte reale Microarray-Daten benutzen, um aus diesen Netzwerke zu generieren. Für diese Graphen wird dann überprüft, wie groß der Anteil der prämoralisierbaren Graphen ist. Es ergibt sich, dass sehr viele der aus Microarray-Daten geschätzten Graphen prämoralisierbar sind. Zudem wird in diesem Abschnitt gezeigt, dass Graphen mit vielen Knoten häufiger nicht prämorlisierbar sind.

In einer Microarray-Studie werden im Regelfall mehr als 10000 verschiedene Gene gemessen. Ein gemeinsames Netzwerk für die Gesamtmenge an Genen zu schätzen ist selbst mit den in Kapitel 2 vorgestellten Algorithmen nicht erfolgsversprechend. Dementsprechend sollte man vor dem Erzeugen des Netzwerkes die Anzahl der Gene und somit die Anzahl der Knoten im Netz reduzieren. Zwei Beispiele für eine mögliche Reduktion werden im Folgenden gegeben und es wird untersucht, wie groß die Anzahl der prämoralisierbaren Graphen ist.

- **Ansatz 1:** Gene werden auf Grund von Wissen, welches sich nicht aus dem Experiment ergibt, zu Gruppen zusammengefasst. Für diese einzelnen Gengruppen wird dann ein Netzwerk aus den Daten geschätzt. In dieser Arbeit wird die KEGG-Datenbank ausgewählt, um mit Hilfe dieser Informationen Gene zu Pathways zusammen zu fassen. Für jede Menge solcher Pathway-Gene wird dann ein Netzwerk geschätzt.

- **Ansatz 2:** Zusätzlich zu externem Wissen wird auch Wissen, welches sich aus dem Experiment selbst ergibt, benutzt, um die Gene zu Gruppen zusammen zu fassen. Es wird zuerst für jeden Datensatz ein Klassifikator erzeugt, welcher zwischen zwei Untergruppen von Proben unterscheiden soll. Die Gene, die für die Klassifikation benutzt werden, werden kombiniert mit den Pathway-Genen aus Ansatz 1. Für die komplette Menge von Genen wird dann ein Netzwerk erzeugt. Die Motivation für ein solches Vorgehen wird in Kapitel 5.4 gegeben.

5.2.1 Ansatz 1: Simulationsaufbau

Für den ersten Ansatz erfolgt die Unterteilung in Gruppen mit Hilfe der KEGG-Datenbank (http://www.genome.jp/kegg/kegg2.html). In der KEGG-Datenbank sind Pathways abgespeichert, wobei jeder Pathway eine komplexe Struktur zwischen einer Menge von Genen beschreibt, so beispielsweise, welche Gene eines Pathways auf welche Art und Weise interagieren. Die aus KEGG gelieferten Informationen werden für diese Studie stark reduziert. Im Folgenden ist ein Pathway definiert als die Menge der Gene, die in dem KEGG-Pathway notiert sind; Interaktionen, die in KEGG abgespeichert sind, werden also nicht berücksichtigt. Dies liegt daran, dass in KEGG auch viele Informationen beschrieben werden, die man auf einem Microarray-Chip gar nicht detektieren kann, wie zum Beispiel Protein-Protein-Interaktion.

Ein Netz bestehend aus Pathway-Genen zu betrachten ist sinnvoll, wenn man davon ausgeht, dass ein Pathway eine funktionelle Einheit ist, in der es zwischen den einzelnen Genen eine große Interaktion gibt, zwischen Genen verschiedener Pathways aber nur eine geringe Interaktion. Wenn es keine Interaktion zwischen Genen aus unterschiedlichen Pathways gäbe, so wäre die bedingte Korrelation zwischen zwei Genen eines Pathways gegeben die übrigen Gene des Pathways die gleiche wie die bedingte Korrelation der zwei Gene gegeben alle anderen Gene. An der bedingten Korrelation gegeben alle anderen Gene ist man natürlich besonders interessiert.

Die Simulation aus Kapitel 5.1 hat gezeigt, dass es sinnvoll ist, Netzwerke auf der Genebene zu erstellen. Dies hat auch Vorteile bei der biologischen Interpretation und Darstellung der Daten (siehe Kapitel 1). Somit wird im Folgenden immer ein Netzwerk auf der Genebene geschätzt. Dazu wird Vorgehen 2 aus Kapitel 5.1 benutzt: falls mehrere Spots ein Gen repräsentieren, so wird jener Spot ausgewählt, der den geringsten Abstand zum Median aus allen Spots aufweist.
Für die Simulation werden neun verschiedene öffentlich zugängige Microarray-Datensätze benutzt. Diese Datensätze umfassen die Bereiche Brusttumore (Gruvberger[32], Wang[75], Nevins[76]), Lungentumore (Bhattacharjee[7], Beer[4], Garber[26]) und auch Herzerkrankung (Hannenhalli[33] und Barth/Kuner A und B[3]). In diesen Datensätzen wurden teilweise unterschiedliche Chiptypen benutzt (siehe Anhang B). Für jeden Datensatz werden Untergruppen der Patienten bezüglich eines vorgegebenen Phänotyps gebildet. Die hier gewählten Untergruppen stellen jeweils eine Unterteilung des Patientenkollektivs dar, die in der jeweiligen Arbeit untersucht worden ist oder generell eine wichtige Rolle bei der Behandlung spielt. So bildet der Faktor, ob ein Patient mit Brustkrebs Estrogen-Rezeptor-positiv oder -negativ ist, eine wichtige Entscheidungshilfe bei der Therapie [65]. Bei Lungenkrebs gibt es die kontrovers diskutierte Hypothese, dass die Tumore, die als *Squamous Cell Carcinoma* klassifiziert werden, eine bessere Prognose haben als Tumore, die als *Adenocarcinoma* klassifiziert werden [14]. Bei Herzerkrankungen wurde gezeigt, dass Patienten mit einer ischämischen Kardiomyopathie(ICM) eine schlechtere Prognose haben als Patienten mit einer dilatativen Kardiomyopathie(DCM) [1]. Somit ist es auch hier von großer Bedeutung die transkriptionellen Unterschiede dieser zwei Patientenkollektive zu kennen, gerade weil die Unterscheidung dieser zwei Erkrankungen mit herkömmlichen Analysen schwierig ist (siehe auch Kapitel 1 und Kapitel 5.4).

Abbildung 5.6 verdeutlicht den weiteren Versuchsaufbau. Für jeden Datensatz werden drei Teildatensätze untersucht, zum einen die beiden Datensätze, die nur aus Proben aus einer der Untergruppen bestehen, und auch der komplette Datensatz mit allen Proben. Dieser Ansatz wird gewählt, weil häufig ein Sinn in der Erstellung von Netzwerken darin besteht, Unterschiede in der Gen-Gen-Interaktion zwischen Patientenuntergruppen zu untersuchen. Aber auch ein Netzwerk auf dem gesamten Datensatz zu erzeugen ist sinnvoll, falls die Hypothese vorliegt, dass es keine Unterschiede zwischen den Untergruppen gibt. Beide Vorgehensweisen werden in Abschnitt näher 5.4 erläutert. Der Datensatz von Nevins[76] nimmt bei dem beschriebenen Vorgehen eine Sonderstellung ein, da hier zwei verschiedene Einteilungen in Probenuntergruppen vorliegen.

Für jeden Pathway aus KEGG und jeden der oben beschriebenen Teildatensätze wird

	Datensatz	Vergleich	Gruppen	Probenzahl	Literatur
1	Wang	Bt: ER Status	Negativ/Positiv	77/209	[75]
2	Hannenhalli	Herzerkrankung	DCM/ICM	86/108	[33]
3	Bhattacharjee	Lt: Subtype	Adeno/Squamous	123/21	[7]
4	Gruvberger	Bt: ER Status	Negativ/Positiv	30/28	[32]
5	Nevins	Bt: LN Status	Negativ/Positiv	23/23	[76]
6	Nevins	Bt: ER Status	Negativ/Positiv	23/23	[76]
7	Beer	Lt	Normal/Tumor	10/86	[4]
8	Garber	Lt: Subtype	Adeno/Squamous	34/12	[26]
9	Barth/Kuner A	Herzerkrankung	DCM/ICM	16/10	[3]
10	Barth/Kuner B	Herzerkrankung	DCM/Normal	13/15	[3]

Tabelle 5.1: Überblick der Datensätze

ein Netzwerk auf Genebene geschätzt. Zuerst wird für jeden einzelnen Datensatz eine Liste erstellt, die für jeden Pathway die Gene enthält, die im Datensatz und im Pathway vorhanden sind. Pathways, für die keine Gene oder nur ein Gen in einem Datensatz vorhanden sind, werden nicht berücksichtigt. Dies führt zu leichten Unterschieden bei der Anzahl der zu untersuchenden Pathways für die verschiedenen Datensätze. Dann wird die Präzessionsmatrix mit Hilfe des in Kapitel 2 beschriebenen Algorithmus von Schäfer und Strimmer geschätzt und bei jedem Eintrag der geschätzten Präzessionsmatrix getestet, ob dieser sich signifikant von Null unterscheidet. Es wird als Ergebnis für jeden Eintrag der Präzessionsmatrix ein gegen multiples Testen korrigierter q-Wert geliefert. Somit ergibt sich ein vollständiger ungerichteter Graph, dessen Knoten Gene repräsentieren. Jede Kante in dem Graphen ist mit dem korrespondierenden q-Wert versehen. Eleminiert man dann alle Kanten, deren q-Wert über einer vorher festgesetzten Schranke liegen, so ergibt sich ein neuer ungerichteter Graph. In diesem werden auch die Knoten entfernt, die nach Reduktion der Kanten keinen Nachbarn mehr haben. Für den so entstandenen Graphen wird mit Hilfe des Algorithmus aus Kapitel 3 getestet, ob dieser Graph prämoralisierbar ist oder nicht.

Die Schranke für den q-Wert wird variiert, es werden die Schrankenwerte 0.01, 0.05, 0.08, 0.1, 0.15 und 0.2 benutzt. Da der Schrankenwert korreliert mit der Dichte des geschätzten Graphen und der Anzahl der Knoten für den Graphen, ist es so möglich, Aussagen darüber zu treffen, ob dichtere Graphen oder Graphen mit mehr Knoten eher prämoralisierbar oder nicht prämoralisierbar sind.

Zusammengefasst wird mit dem obigen Vorgehen ein Interaktionsnetzwerk für eine Untermenge von Genen und Proben geschätzt und dann überprüft, ob der so geschätzte Graph prämoralisierbar ist oder nicht. Unter der Prämisse, dass die Schätzung des Interaktionsnetzwerkes die reale Interaktion widerspiegelt, wird mit dieser Studie überprüft, wie groß der Anteil der prämoralisierbaren Graphen bei Gen-Gen-Interaktionsgraphen ist. Da der Algorithmus, welcher aus einem prämoralisierbaren aber nicht zerlegbaren Graphen eine zufällige Matrix mit den geforderten Eigenschaften erzeugt, sehr viel rechenintensiver ist als der Algorithmus für zerlegbare Graphen, ist es auch von großem Interesse zu sehen, wie

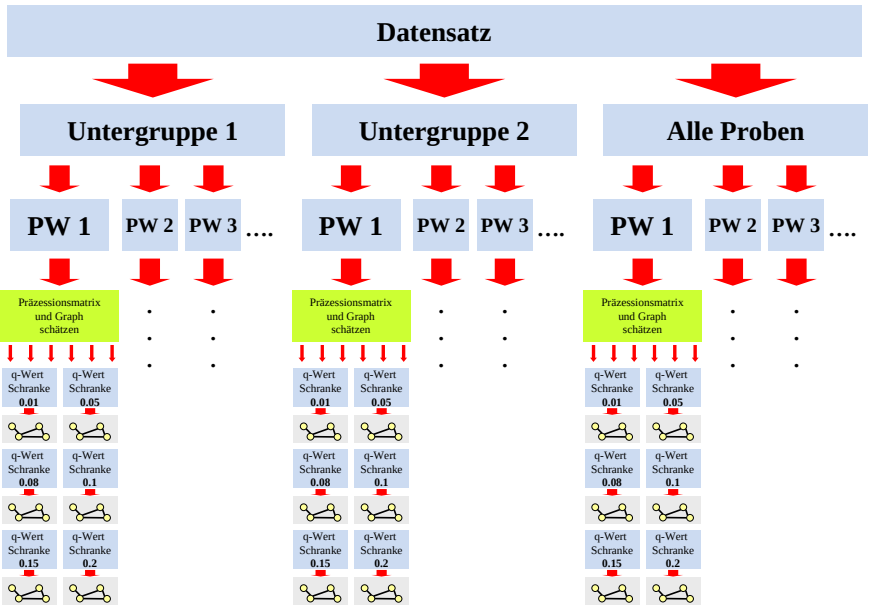

Abbildung 5.6: Schema des Versuchsaufbaus in Ansatz 1.

groß der Anteil der zerlegbaren Graphen an den prämoralisierbaren Graphen ist.

Die resultierenden Graphen werden in drei Gruppen unterteilt. In Gruppe 0 befinden sich die Graphen, bei denen bei der Schätzung der Präzessionsmatrix oder beim Testen auf Signifikanz ein Fehler in der Berechnung aufgetreten ist. In Gruppe 1 befinden sich die Graphen, die entweder keine Kante haben, wo also der kleinste q-Wert der zugehörigen Präzessionsmatrix größer ist als der angesetzte Schrankenwert, oder für die die maximale Anzahl von Nachbarn 1 ist. Es ist sinnvoll, solche Graphen getrennt zu betrachten, weil jede Zusammenhangskomponente eines solchen Graphen selbst ein vollständiger Graph ist und es somit keine Probleme bei der Erzeugung der Matrix gibt, da man für jede Zusammenhangskomponente keine Restriktionen hat. Alle übrigen Graphen gehören zu Gruppe 2. Für jeden Graphen aus Gruppe 2 wird mit Hilfe des in Kapitel 3 beschriebenen Algorithmus überprüft, ob der Graph prämoralisierbar ist oder nicht. Somit unterteilt sich Gruppe 2 in

- **Gruppe 2.A:** Menge der prämoralisierbaren Graphen aus Gruppe 2
- **Gruppe 2.B:** Menge der nicht prämoralisierbaren Graphen aus Gruppe 2

5.2.2 Ansatz 1: Ergebnisse

Es gibt 10 Vergleiche. Bei jedem Vergleich wird das Netzwerk auf jeweils drei Teildatensätzen geschätzt und für jeden Teildatensatz gibt es 6 verschiedene Schranken für den q-Wert. Mit ungefähr 172 zu testenden Pathways in jedem Vergleich ergibt dies zusammen 31068

Graphen. In Gruppe 0 befinden sich 8034 Graphen, das bedeutet bei 8034 Untersuchungen gab es entweder beim Schätzen der Präzessionsmatrix oder beim Testen auf Signifikanz einen Fehler in der Berechnung. Dies betrifft vor allem Pathways mit sehr wenigen Genen (siehe Abbildung 5.7).

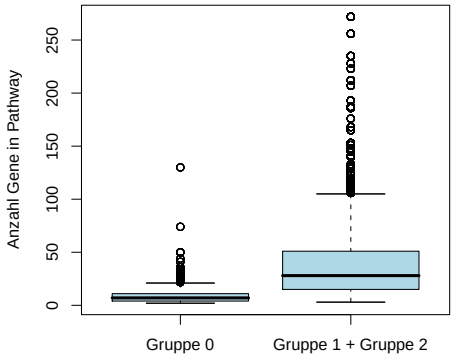

Abbildung 5.7: Verteilung der Anzahl der Knoten für Gruppe 0 und die Gruppen 1 und 2

In Gruppe 1 befinden sich 12534 Graphen mit keiner Kante und 3934 Graphen mit maximal einem Nachbarn, also insgesamt 16468 Graphen. Jeder Graph aus dieser Gruppe ist prämoralisierbar, denn jeder Graph ist zerlegbar.

Die Anzahl der Graphen aus Gruppe 2 beträgt 6566. In Gruppe 2 liegt der Anteil der prämoralisierbaren Graphen bei 73%, das heißt 4802 der 6566 Graphen gehören zu Gruppe 2.A. Fasst man Gruppe 1 und Gruppe 2 zusammen, so ergibt sich, dass 21270 der 23034 Graphen prämoralisierbar sind, was einem Anteil von 92% entspricht. Die Zusammensetzung von prämoralisierbaren Graphen und zerlegbaren Graphen in Gruppe 2 wird in Tabelle 5.2 gezeigt. Wie in Kapitel 3 bewiesen sind alle zerlegbaren Graphen prämoralisierbar. Es gibt auch einige prämoralisierbare Graphen, die nicht zerlegbar sind (7 %). Vier dieser Graphen sind in Abbildung 5.8 dargestellt. Man erkennt deutlich, dass es bei jedem dieser Graphen einen Zykel ohne Abkürzung gibt und dass es bei jedem Zykel ohne Abkürzung zwei benachbarte Knoten mit gemeinsamen Nachbarn gibt. Das ist eine notwendige Bedingung dafür, dass ein nicht zerlegbarer Graph prämoralisierbar ist (Satz 3.1.6).

Im Folgenden wird untersucht, ob es Einflussfaktoren gibt, die bestimmen, ob ein Graph prämoralisierbar ist oder nicht. Dies geschieht zuerst auf deskriptiver Ebene. Für alle Untersuchungen bezieht man sich nur auf die Graphen aus Gruppe 2.A und 2.B, da alle Graphen aus Gruppe 1 zerlegbar sind und es somit keine Korrelation zwischen den Parametern und der Prämoralisierungseigenschaft gibt.

Zuerst betrachtet man die einzelnen Vergleiche und Datensätze, die in die Auswertung eingegangen sind, getrennt voneinander. Tabelle 5.3 zeigt die Anzahl der prämoralisierbaren

	nicht zerlegbar	zerlegbar
nicht prämoralisierbar	1764	0
prämoralisierbar	321	4481

Tabelle 5.2: Tabelle der prämoralisierbaren und nicht prämoralisierbaren Graphen aus der Gruppe 2 für Ansatz 1

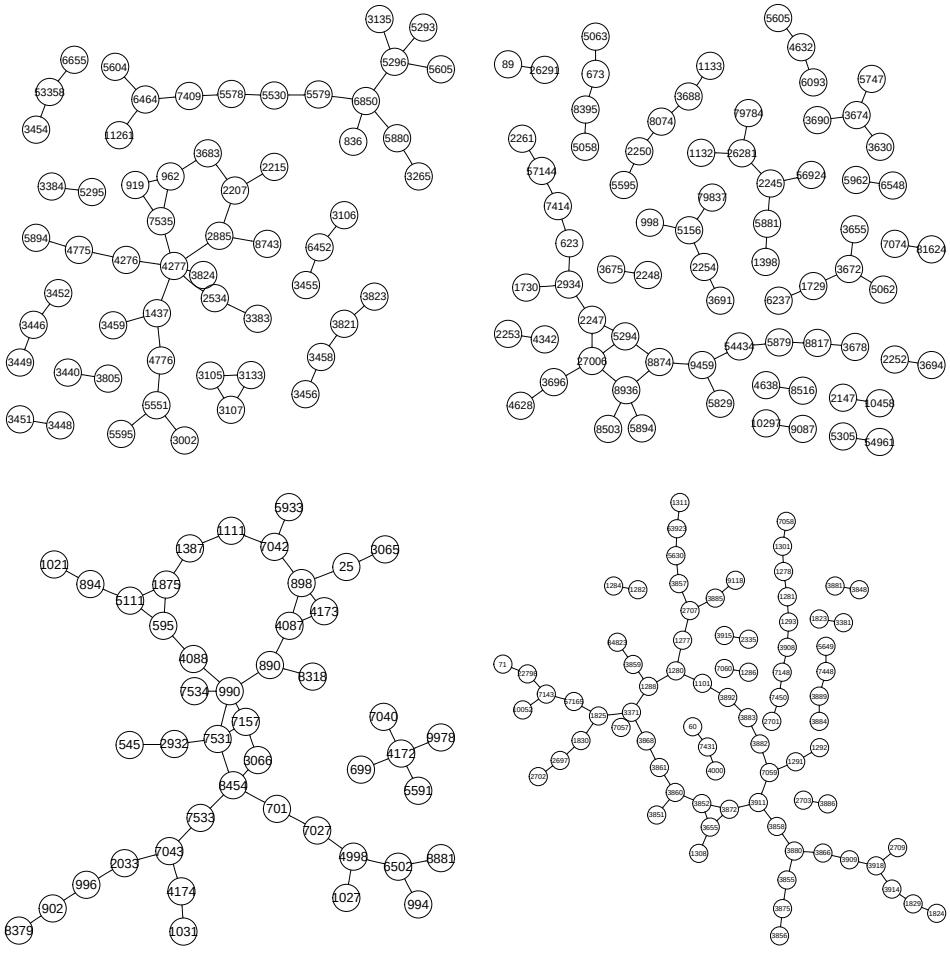

Abbildung 5.8: Nicht zerlegbare aber prämoralisierbare Graphen aus Microarray-Daten geschätzt.

	Datensatz	Vergleich	Anzahl Gr. 2.A	Anzahl Gr. 2.B
1	Wang	Bt: ER Status	1057	280
2	Hannenhalli	Herzerkrankung	780	186
3	Bhattacharjee	Lt: Subtype	706	181
4	Gruvberger	Bt: ER Status	482	223
5	Nevins	Bt: LN Status	325	150
6	Nevins	Bt: ER Status	319	141
7	Beer	Lt	529	296
8	Garber	Lt: Subtype	205	75
9	Barth/Kuner A	Herzerkrankung	168	109
10	Barth/Kuner B	Herzerkrankung	231	123

Tabelle 5.3: Anzahl der Graphen aus Gruppe 2.A und 2.B für verschiedene Datensätze

Graphen aufgeteilt nach den einzelnen Vergleichen. Man erkennt, dass in allen Datensätzen viele prämoralisierbare Graphen geschätzt worden sind. Das bedeutet, prämoralisierbare Graphen existieren nicht nur in einigen Datensätzen und insbesondere nicht nur für eine Chip-Technologie (cDNA, Oligo). Weiterhin ist aus Tabelle 5.3 ersichtlich, dass bei Datensätzen mit einer großen Gesamtmenge von vorhandenen Proben die Anzahl der Graphen aus der Gruppe 2 höher ist als bei Datensätzen mit weniger Proben. Es werden in diesen Datensätzen also komplexere graphische Strukturen erzeugt. Auch der Anteil der prämoralisierbaren Graphen (Gruppe 2.A) ist bei diesen Datensätzen größer, vor allem bei Wang, Hannenhalli und Bhattacharjee.

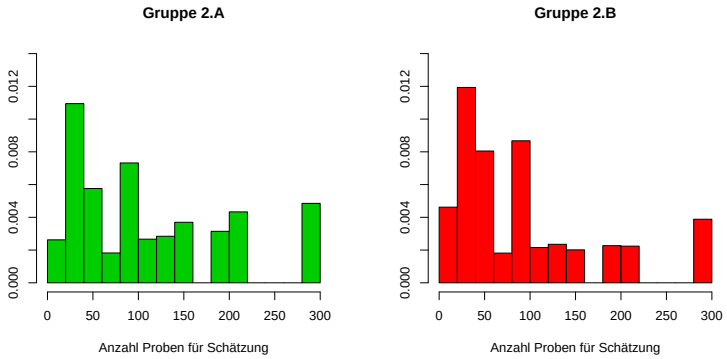

Abbildung 5.9: Für die prämoralisierbaren Graphen und die nicht prämoralisierbaren Graphen ist die Anzahl der in die Schätzung eingegangenen Proben aufgetragen.

Dies lässt vermuten, dass die Prämoralisierungseigenschaft von der Anzahl der in die Schätzung eingehenden Beobachtungen abhängen könnte. Aber in den Vergleichen werden nicht nur Graphen auf dem gesamten Probensatz geschätzt, sondern auch auf Untergruppen, die sich in ihrer Größe zwischen den Vergleichen stark unterscheiden (siehe Abbildung

5.1). Um also der obigen Vermutung nachzugehen, wird unabhängig vom Datensatz die Anzahl der in die Schätzungen der einzelnen Graphen eingehenden Proben getrennt für Graphen aus Gruppe 2.A und 2.B aufgetragen (Abbildung 5.9). Man erkennt hier, dass sich die oben formulierte Hypothese nicht bestärken lässt. Es ist kein klarer Unterschied zwischen prämoralisierbaren und nicht prämoralisierbaren Graphen zu erkennen. Also ist keine klare Tendenz auszumachen, dass bei einer größeren Probenzahl mehr prämoralisierbare Graphen erzeugt werden.

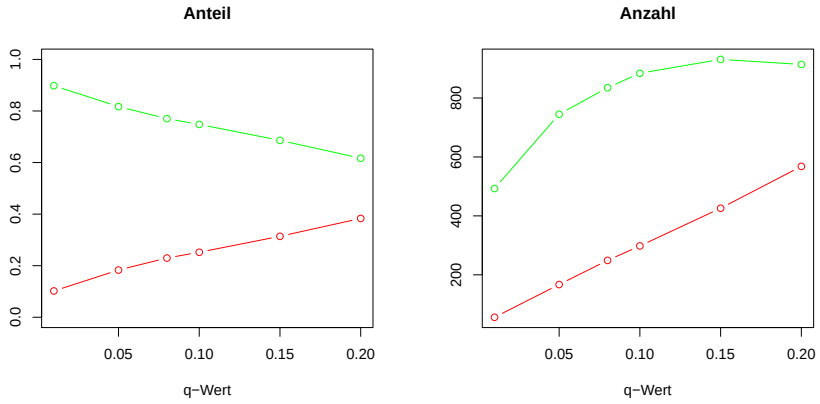

Abbildung 5.10: Für die verschiedenen q-Wert-Schranken ist der Anteil und die Anzahl der prämoralisierbaren (grün) und nicht prämoralisierbaren (rot) Graphen aus Gruppe 2 aufgetragen.

Wie beschrieben werden in dem vorgestellten Ansatz verschiedene Schranken für den q-Wert benutzt, wobei eine höhere Schranke einen Graphen mit mehr Kanten und Knoten zur Folge hat. Deshalb wird die Prämoralisierungseigenschaft in Abhängigkeit dieser Schranke untersucht. In Abbildung 5.10 ist der Anteil und die Anzahl der prämoralisierbaren und nicht prämoralisierbaren Graphen an allen erzeugten Graphen bei verschiedenen Schranken für den q-Wert aufgetragen. Man erkennt, dass die Anzahl der nicht prämoralisierbaren Graphen mit steigender q-Wert-Schranke ansteigt, während die Anzahl der prämoralisierbaren Graphen für hohe q-Wert-Schranken fast konstant ist. Somit sinkt der Anteil der prämoralisierbaren Graphen mit steigender Schranke. Zwei mögliche Szenarien, die in einer solchen Grafik resultieren, sind:

1. Die Graphen, die bei einer niedrigen Schranke für den q-Wert prämoralisierbar sind, sind auch bei einer höheren Schranke prämoralisierbar, und es kommen nicht prämoralisierbare Graphen bei Benutzung einer höheren Schranke hinzu, die vorher der Gruppe 1 oder 0 angehört haben.

2. Ein Teil der Graphen, die bei einer niedrigen Schranke prämoralisierbar sind, sind bei einer höheren Schranke nicht mehr prämoralisierbar und es kommen prämorali-

sierbare Graphen bei Benutzung der höheren Schranke hinzu, die vorher der Gruppe 1 oder 0 angehört haben.

Um zu untersuchen, welches Szenario zutreffender ist, betrachtet man die Schranken 0.1 und 0.2. Durch Erhöhung der Schranke für den q-Wert werden nur Kanten und eventuell Knoten in einen Graphen hinzugefügt, es wird aber keine Kante und kein Knoten gelöscht. Somit erhält man für jede Kombination aus Datensatz, Vergleich, Untergruppe und Pathway eine Folge von zwei Graphen, wobei der erste Graph ein Teilgraph des zweiten Graphen ist. Für jede dieser Kombinationen betrachtet man nun, ob ein Graph, welcher bei einer Schranke von 0.1 nicht prämoralisierbar ist, durch Erhöhung der Schranke auf 0.2 prämoralisierbar wird oder andersherum.

Insgesamt gibt es 1494 zu untersuchende Kombinationen. Die geringe Zahl ist darin begründet, dass für viele der ursprünglichen Kombinationen Graphen geschätzt worden sind, die für beide Schranken der Gruppe 0 oder 1 angehören. Es werden aber nur solche Kombinationen beachtet, wo mindestens ein Graph der Gruppe 2 angehört. Das Ergebnis der Untersuchung zeigt Tabelle 5.4 und ist in Abbildung 5.11 grafisch dargestellt. Man erkennt deutlich, dass Graphen, die bei einer q-Wert-Schranke von 0.1 zu Gruppe 0 oder 1 gehören, aber bei einer Schranke von 0.2 zu Gruppe 2, fast ausschließlich in die Gruppe 2.A fallen, also prämoralisierbar sind. Hingegen gibt es einen großen Teil von Graphen, die bei einer Schranke von 0.1 prämoralisierbar sind, also zu Gruppe 2.A gehören, bei einer Schranke von 0.2 aber nicht mehr prämoralisierbar sind, also zur Gruppe 2.B gehören. Szenario 2 ist demnach realistischer.

	Gruppe 0 und 1	Gruppe 2.A	Gruppe 2.B
Gruppe 0 und 1	-	291	21
Gruppe 2.A	10	603	271
Gruppe 2.B	2	20	276

Tabelle 5.4: Ergebnisse der Untersuchungen bei verschiedenen q-Wert-Schranken. In den Zeilen ist die Zugehörigkeit bei einer Schranke von 0.1 aufgetragen und in den Spalten die Zugehörigkeit bei einer Schranke von 0.2

Graphen, die bei einer hohen Schranke neu in die Gruppe 2 eintreten, also vorher keine oder nur sehr wenige Kanten gehabt haben, besitzen tendenziell weniger Knoten und weniger Kanten als Graphen, die sowohl bei einer Schranke von 0.1 als auch bei einer Schranke von 0.2 zur Gruppe 2 gehören (Abbildung 5.12). Also scheinen vor allem Graphen mit wenig Knoten oder aber wenigen Kanten prämoralisierbar zu sein. Zur Überprüfung dieser These betrachtet man die Anzahl der Knoten zwischen den prämoralisierbaren und den nicht prämoralisierbaren Graphen aus Gruppe 2 unabhängig von der Schranke für den q-Wert. Jeder Knoten hat hier mindestens einen Nachbarn.

In Abbildung 5.13 ist die empirische Verteilung der Anzahl der Knoten mit Nachbarn bei prämoralisierbaren Graphen (Gruppe 2.A) und nicht prämoralisierbaren Graphen (Gruppe

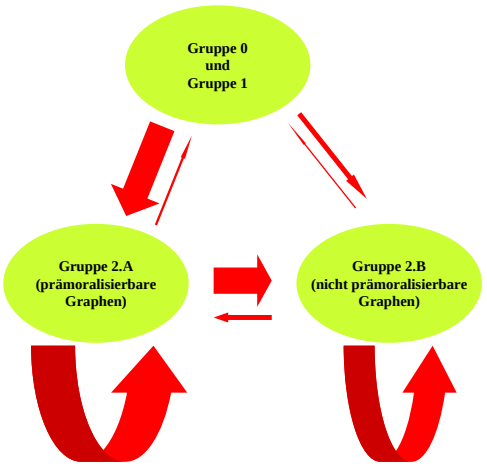

Abbildung 5.11: Schema der Gruppenzugehörigkeit für die Schrankenwerte 0.1 und 0.2. Der Pfeil verläuft von der Zugehörigkeit bei einer Schranke von 0.1 zu der Zugehörigkeit bei einer Schranke von 0.2. Die Dicke des Pfeiles repräsentiert die Anzahl der jeweiligen Graphen.

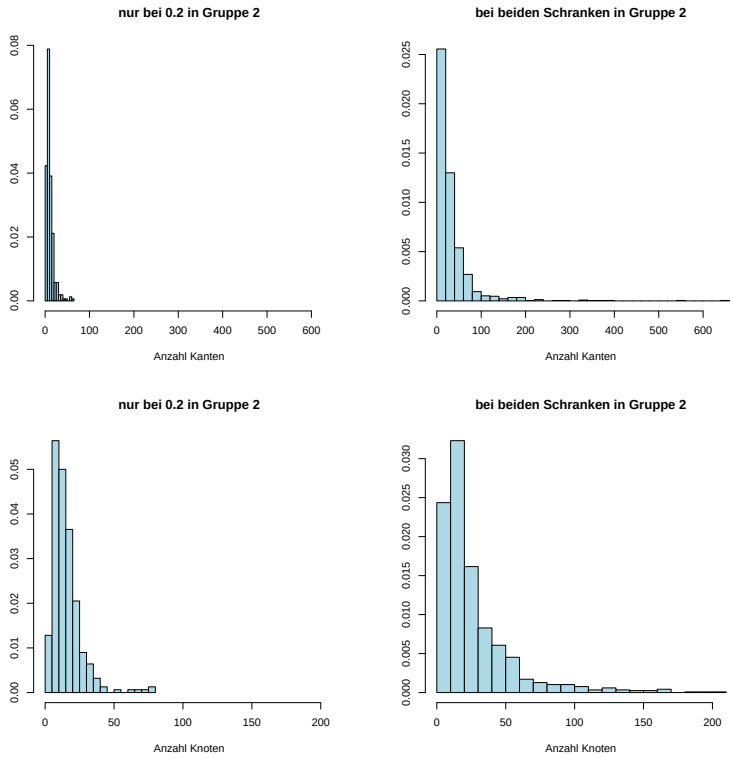

Abbildung 5.12: Anzahl der Knoten und Kanten für zwei verschiedene Schranken des q-Wertes (0.1 und 0.2)

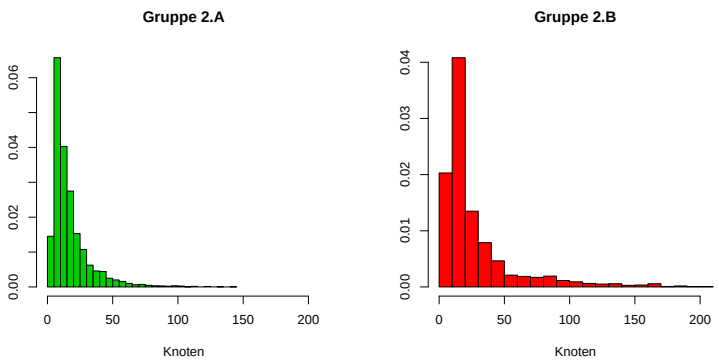

Abbildung 5.13: Für die prämoralisierbaren Graphen (Gruppe 2.A) und die nicht prämoralisierbaren Graphen (Gruppe 2.B) ist die Anzahl der Knoten aufgetragen.

2.B) aufgetragen. Wie zu erwarten ist die Anzahl der Knoten für prämoralisierbare Graphen kleiner als bei nicht prämoralisierbaren Graphen.

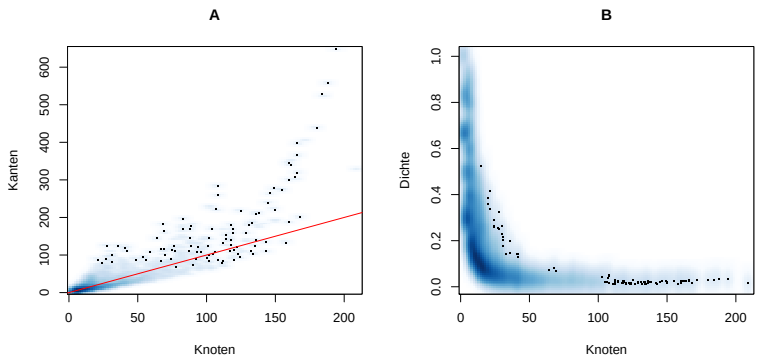

Abbildung 5.14: Für die Graphen aus Gruppe 2 ist die Anzahl der Knoten gegenüber der Anzahl der Kanten (A) beziehungsweise gegenüber der Dichte des Graphen (B) aufgetragen.

Wenn man die Anzahl der Kanten betrachtet, so ergibt sich, dass diese in Gruppe 2 mit der Anzahl der Knoten einen fast linearen Zusammenhang bilden (Abbildung 5.14, A). Somit wird sich bei Betrachtung der Kanten kein anderes Bild ergeben als bei der Betrachtung der Knoten. Interessanter ist es, die Dichte des Graphen zu betrachten.

Definition 5.2.1 *Sei Graph $H = (V_H, E_H)$ ein Graph. Die Dichte von H ist definiert als*

$$d(H) := \frac{\#E_H}{\binom{\#V_H}{2}}.$$

Ein vollständiger Graph H mit n Knoten besitzt genau $\binom{n}{2}$ Kanten und hat demnach eine Dichte von 1. Für einen nicht vollständigen Graphen H mit n Knoten ist die Dichte immer kleiner als 1, und gibt den Anteil der in H vorhandenen Kanten an allen denkbaren Kanten in H an, das sind die Kanten, die sich in dem vollständigen Graphen mit n Knoten befinden.

Zwar gibt es auch für die Dichte einen starken Zusammenhang zwischen Anzahl der Knoten und Anzahl der Kanten, allerdings nur im Bereich mit wenigen Knoten (Abbildung 5.14, B). Es fällt zudem auf, dass ab ca. 50 Knoten die Dichte der Graphen relativ konstant ist. Die empirische Verteilung der Dichte der Graphen ist für Graphen der Gruppen 2.A und 2.B in Abbildung 5.15 dargestellt. Es sind hier wenige Unterschiede erkennbar. Insgesamt ist die Dichte bei den geschätzten Graphen gering. Dies ist nicht verwunderlich und unterstützt die Behauptung, dass die Interaktionsgraphen zwischen Genen *sparse* sind, also nur wenige Kanten zwischen den Knoten existieren.

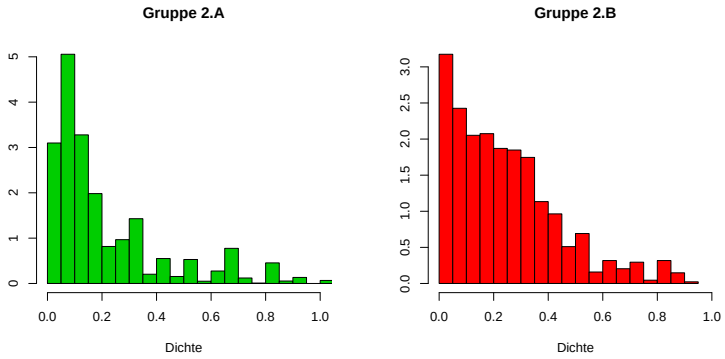

Abbildung 5.15: Für die prämoralisierbaren (Gruppe 2.A) Graphen und die nicht prämoralisierbaren Graphen (Gruppe 2.B) ist die Dichte dargestellt.

Insgesamt zeigen die ersten Untersuchungen, dass der Anteil der prämoralisierbaren Graphen an allen erzeugten Graphen sehr hoch ist. Es zeigt sich aber auch, dass die Prämoralisierungseigenschaft mit anderen Eigenschaften eines Graphen, wie der Anzahl der Knoten, korreliert. Dies ist bis jetzt nur deskriptiv gezeigt worden und soll nun mit Hilfe eines Tests belegt werden. Im Detail wird getestet, ob sich die Anzahl der Knoten zwischen prämoralisierbaren und nicht prämoralisierbaren Graphen unterscheidet und ob sich die Verteilungen der Dichte eines Graphen zwischen prämoralisierbaren und nicht prämoralisierbaren Graphen unterscheidet. Abbildung 5.16 zeigt, dass es Unterschiede bei der Knotenverteilung und der Graphendichte über die Datensätze hinweg gibt. Aus diesem Grund wird in dem folgenden Test ein Modell benutzt, in dem der *Datensatz*-Parameter als zufälliger Effekt Berücksichtigung findet.

Will man testen, ob sich die beiden Verteilungen zwischen den prämoralisierbaren und nicht prämoralisierbaren Graphen unterscheiden, so ist dies mit dem Problem behaftet, dass die einzelnen zu Grunde liegenden Beobachtungen nicht unabhängig sind. Dies ist

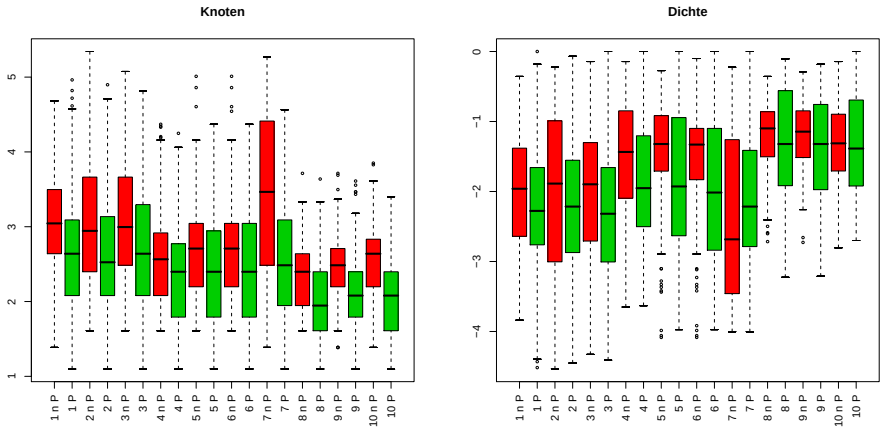

Abbildung 5.16: Für jeden Vergleich werden für die prämoralisierbaren (grün) und nicht prämoralisierbaren (rot) Graphen die logarithmierte Verteilung der Knoten und die logarithmierte Verteilung der Graphendichte dargestellt. Die Reihenfolge der Vergleiche entspricht Tabelle 5.1

darin begründet, dass sich die Pathways überschneiden, das bedeutet, es gibt eine Untermenge von Genen, die in verschiedenen Pathways auftreten. Zudem sind verschiedene q-Werte-Schranken benutzt worden, das bedeutet, der gleiche Pathway wurde 6 Mal untersucht, was eine zusätzliche Abhängigkeit ergibt. Um diese Abhängigkeiten zu umgehen und trotzdem einen Test zu machen, werden für das zu untersuchende Kollektiv folgende Einschränkungen gemacht. Zum einen werden die einzelnen Schranken für den q-Wert getrennt betrachtet, zum anderen werden nur solche Beobachtungen benutzt, bei denen die zu Grunde liegenden Pathways paarweise disjunkt sind. So erhält man zwar nur eine kleinere Menge von Beobachtungen, aber diese sind unabhängig. Es werden zudem die Graphen entfernt, die auf dem gesamten Datensatz erzeugt worden sind.

Zum Erzeugen der unabhängigen Pathways wird für jede q-Wert-Schranke zufällig aus der Menge der Pathways gezogen und ein Pathway wird gewählt, falls er disjunkt ist zu allen vorher gewählten Pathways. Der Vorgang wird so lange wiederholt, bis 50 verschiedene Pathways gezogen worden sind. Auf dem gezogenen Kollektiv von disjunkten Pathways wird dann für die Knotenanzahl das folgende *mixed-effects*-Modell an die Daten angepasst:

$$log(Knoten) = intercept + Datensatz + praemoralisierbar + \epsilon$$

Hierbei ist $Datensatz \sim N(0, \sigma_d^2)$ and $\epsilon \sim N(0, \sigma^2)$. Dieses Modell wird per ANOVA mit dem restringierten Modell

$$log(Knoten) = intercept + Datensatz + \epsilon$$

verglichen, um einen p-Wert für die Signifikanz des *praemoralisierbar* Parameters zu ermitteln. Der komplette Vorgang wird mit 1000 Realisierungen disjunkter Pathways wiederholt,

um Schwankungen beim Erzeugen der p-Werte aufzuzeigen. Der Test für die Dichte verläuft analog.

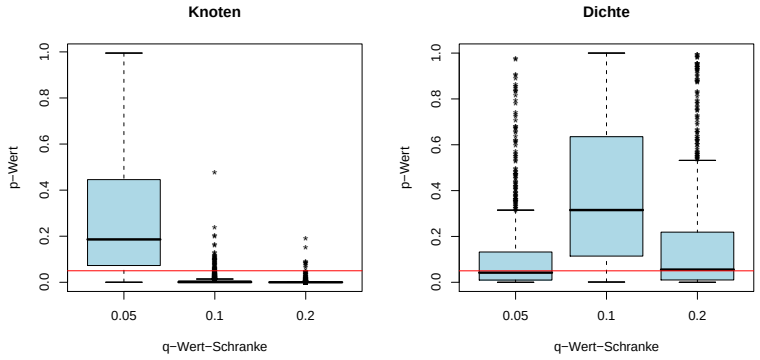

Abbildung 5.17: Verteilung der p-Werte für den Test auf Dichte und auf Knotenanzahl. Eine horizontale Linie ist bei 0.05 eingezeichnet.

Abbildung 5.17 zeigt die Schwankung der auf obige Weise erzeugten p-Werte für die drei untersuchten q-Wert-Schranken 0.05, 0.1 und 0.2. Tabelle 5.5 zeigt die mittleren p-Werte samt Standardabweichung für die Tests. Zusätzlich ist hier die Anzahl der prämoralisierbaren und nicht prämoralisierbaren Graphen eingetragen. Trotz teilweise großer Schwankungen bei den p-Werten zeigt sich, dass zumindest für die Schrankenwerte 0.1 und 0.2 ein signifikanter Unterschied in der Verteilung der Knoten zwischen prämoralisierbaren und nicht prämoralisierbaren Graphen besteht. Für die Dichte zeigt sich, dass im Mittel der Dichteparameter für keine q-Wert-Schranke signifikant ist.

	Gr 2.A	Gr 2.B	p Knoten	SD	p Dichte	SD
0.05	119	28	0.2857	0.2681	0.1090	0.1651
0.1	132	46	0.0099	0.0282	0.3823	0.2974
0.2	142	85	0.0026	0.0120	0.1587	0.2250

Tabelle 5.5: Ergebnisse der Knoten- und Dichtetests

Die durch den Test erhaltenen Resultate bestätigen die Thesen, die man aus dem deskriptiven Teil aufgestellt hat. Die Anzahl der Knoten steht in einem Zusammenhang mit der Prämoralisierungseigenschaft, das bedeutet, wenn ein Graph, der aus Microarray-Daten geschätzt worden ist, viele Knoten hat, so ist die Wahrscheinlichkeit geringer, dass dieser prämoralisierbar ist. Diese Tendenz wird man auch in dem folgenden Ansatz 2 wieder finden, in dem die einzelnen zu untersuchenden Graphen mehr Gene haben als in Ansatz 1.

5.2.3 Ansatz 2: Simulationsaufbau

Ausgangspunkt sind auch hier die aus der vorherigen Analyse bekannten Datensätze. Für jeden dieser Datensätze wurden die in Tabelle 5.1 angegebenen Vergleiche als Grundlage einer Klassifikation mit PAM[71] benutzt. Um die Fehlklassifikationsrate zu schätzen, wurde zuerst 20 Mal eine 10-fache Kreuzvalidierung durchgeführt. Somit wird 200 Mal eine zufällige Untermenge (Trainingsset) des gesamten Datensatzes gewählt, wobei bei der Auswahl sicher gestellt wird, dass das Verhältnis der beiden Gruppen in der Untermenge dem Verhältnis im gesamten Datensatz entspricht (*balanced folds*). Auf jedem Trainings-Datensatz wird für den PAM Algorithmus der Parameter *threshold* optimiert (aus einer Liste mit 20 möglichen Werten), welcher die Stärke der Regularisierung bestimmt, und zusätzlich wird eine initiale, auf Varianz beruhende Filterung durchgeführt, welches eine Liste von 500 möglichen Genen liefert. Der ganze Vorgang wird in [59] detailliert beschrieben.

Für die Klassifikationen, die eine gute Fehlklassifikationsrate liefern, wird dann ein finaler PAM Klassifikator auf dem gesamten Datensatz erzeugt und die Entrez IDs, welche in diesem Klassifikator enthalten sind, werden notiert. Dies sind die Klassifikations-Gene. Diese Gene werden mit den aus KEGG ausgewählten Pathway-Genen kombiniert und es wird ein gemeinsames Netzwerk geschätzt, so wie in Kapitel 5.2, Ansatz 1 beschrieben worden ist. In diesem Fall wird allerdings nur ein Netzwerk auf den einzelnen Unter-Patientengruppen geschätzt, nicht aber auf dem gesamten Datensatz. Die resultierenden Graphen werden, wie im letzten Abschnitt, in die Gruppen 0, 1 und 2 eingeteilt, wobei sich Gruppe 2 noch in die Menge der prämoralisierbaren (Gruppe 2.A) und nicht prämoralisierbaren (Gruppe 2.B) Graphen unterteilt.

5.2.4 Ansatz 2: Ergebnisse

Die Ergebnisse der Schätzung der Fehlklassifikationsrate sind in Abbildung 5.18 dargestellt. Man erkennt, dass die Klassifikation im Datensatz von Hannenhalli und im Barth/Kuner A Datensatz keine guten Ergebnisse liefert. Bei beiden Vergleichen handelt es sich um die Unterscheidung zwischen dilatativer und ischämischer Kardiomyopathie, welche, wie in Kapitel 1 erläutert, als sehr schwierig angesehen wird. Diese zwei Datensätze werden nicht in den folgenden Untersuchungen verwendet. Ebenso geht der Datensatz von Nevins nicht in die weiteren Untersuchungen ein, da das Ergebnis beim LN Vergleich sehr schlecht ist, beim Vergleich mit dem ER Status das Ergebnis ebenfalls nicht befriedigend ist und zusätzlich bei diesem Vergleich der finale Klassifikator nur zwei Gene umfasst (siehe Tabelle 5.6).

Nach der Reduktion auf die übrigen Datensätze verbleiben noch 12384 Graphen für die folgenden Untersuchungen. In Gruppe 0 befinden sich 42 Graphen. Diese Zahl ist sehr viel kleiner als die Anzahl der Elemente in Gruppe 0 in Ansatz 1. In Ansatz 1 hatte man gesehen, dass Fehler vor allem bei Genmengen kleiner Größe auftreten. In diesem Ansatz ist die Anzahl der Gene aber immer größer, da zu jedem Pathway noch die Klassifikations-Gene hinzu genommen werden. Deshalb treten nur noch wenige Fehler auf. In Gruppe 1 befinden sich 3587 Graphen mit keiner Kante und 1937 Graphen mit maximal einem Nachbarn, also

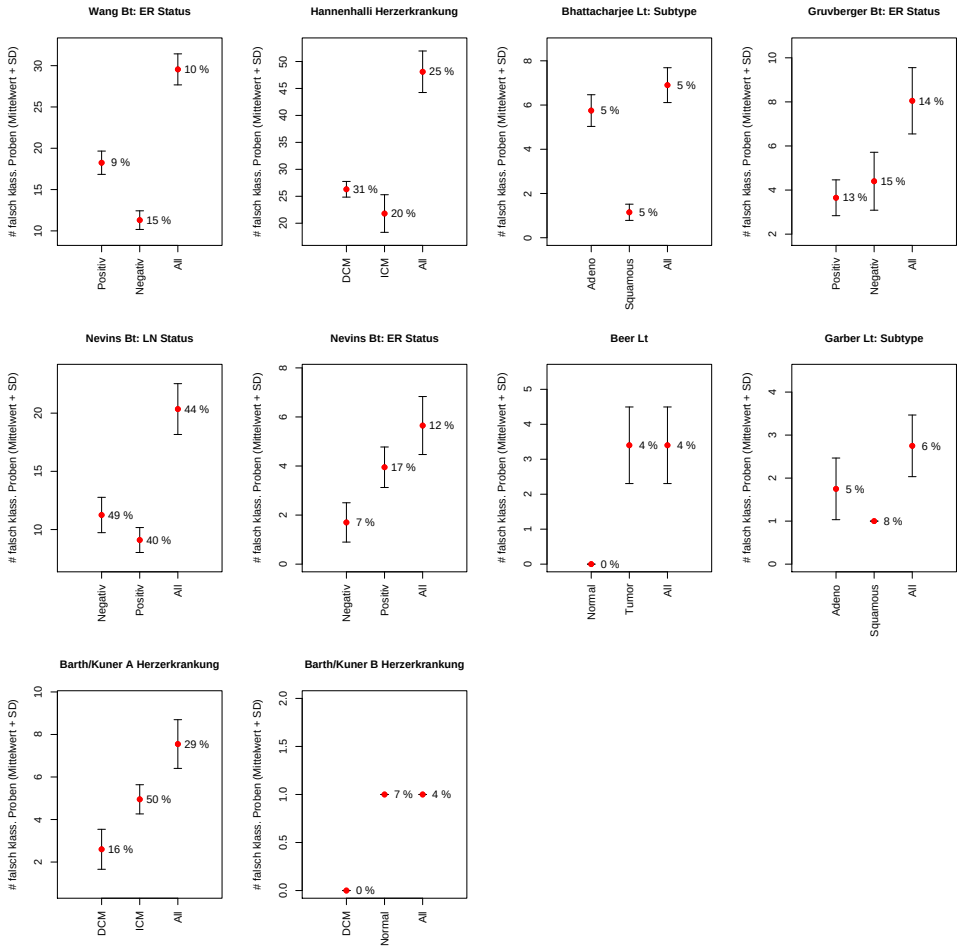

Abbildung 5.18: Klassifikationsresultate: Für jeden Datensatz ist Mittelwert und Standardabweichung der Anzahl der falsch klassifizierten Proben aufgetragen, einmal bezüglich jeder Untergruppe aber auch insgesamt. Die Werte werden auf Grund der 20 Wiederholungen der Kreuzvalidierung erzeugt.

	Data	Vergleich	Anzahl
1	Wang	Bt: ER Status	133
2	Hannenhalli	Herzerkrankung	74
3	Bhattacharjee	Lt: Subtype	133
4	Gruvberger	Bt: ER Status	113
5	Nevins	Bt: LN Status	183
6	Nevins	Bt: ER Status	2
7	Beer	Lt	18
8	Garber	Lt: Subtype	30
9	Barth/Kuner A	Herzerkrankung	88
10	Barth/Kuner B	Herzerkrankung	41

Tabelle 5.6: Anzahl der Klassifikations-Gene bei PAM

insgesamt 5524 Graphen. Alle Graphen aus dieser Gruppe sind prämoralisierbar.

		nicht zerlegbar	zerlegbar
nicht	prämoralisierbar	3286	0
	prämoralisierbar	51	3481

Tabelle 5.7: Tabelle der prämoralisierbaren und nicht prämoralisierbaren Graphen aus der Gruppe 2 für Ansatz 2

Die Anzahl der Graphen aus Gruppe 2 beträgt 6818. In Gruppe 2 liegt der Anteil der prämoralisierbaren Graphen bei 52%, das bedeutet Gruppe 2.A enthält 3532 Graphen und Gruppe 2.B enthält 3286 Graphen. Die Zusammensetzung von prämoralisierbaren Graphen und zerlegbaren Graphen wird in Tabelle 5.7 gezeigt. Der Anteil der prämoralisierbaren Graphen, die nicht zerlegbar sind, liegt hier bei nur 1 %. Zusammengefasst ist der Anteil der prämoralisierbaren Graphen im zweiten Ansatz deutlich geringer als im ersten Ansatz. Da dort gezeigt worden ist, dass die Prämoralisierungseigenschaft mit der Anzahl der Knoten korreliert und beim Klassifikationsansatz in jedem Graphen die Anzahl der Knoten höher ist als beim entsprechenden Graphen in Ansatz 1 (Abbildung 5.19), ist dies nicht verwunderlich.

Die Verteilungen der Knotenanzahl für die Gruppe 2.A und die Gruppe 2.B zeigt Abbildung 5.20. Wie in Ansatz 1 ist ein deutlicher Unterschied zwischen den Verteilungen zu erkennen. Abbildung 5.21 (A) demonstriert, dass es auch in diesem Ansatz eine Korrelation zwischen der Anzahl der Knoten und der Anzahl der Kanten gibt, so dass es auch in diesem Fall besser ist, die Dichte der Graphen zu betrachten. Bei der Dichte zeigt sich bis zu einer Knotenanzahl von ungefähr 50 eine Korrelation zur Knotenanzahl. Für Werte zwischen 50 und 150 scheint es zwei Kollektive von Graphen zu geben, die sich in der Dichte unterscheiden. Die Verteilung der Dichte aufgeteilt nach Gruppe 2.A und Gruppe 2.B (Abbildung 5.22) legt die Vermutung nahe, dass diese zwei Kollektive bei den nicht prämoralisierbaren

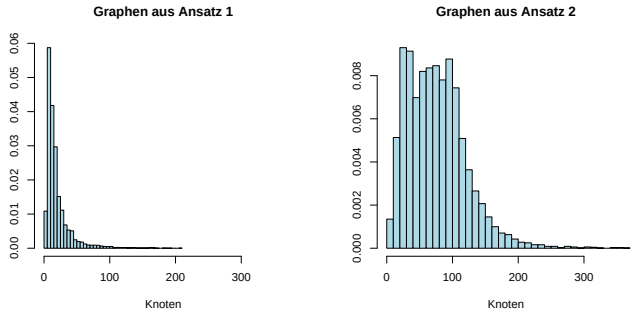

Abbildung 5.19: Für die Graphen aus Ansatz 1, die der dortigen Gruppe 2 angehören, und die Graphen aus Ansatz 2, die der dortigen Gruppe 2 angehören, werden die Verteilungen der Knotenanzahl verglichen.

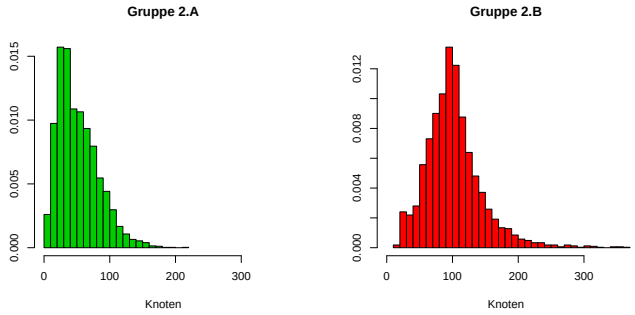

Abbildung 5.20: Für die prämoralisierbaren Graphen (Gruppe 2.A) und die nicht prämoralisierbaren Graphen (Gruppe 2.B) ist die Anzahl der Knoten dargestellt.

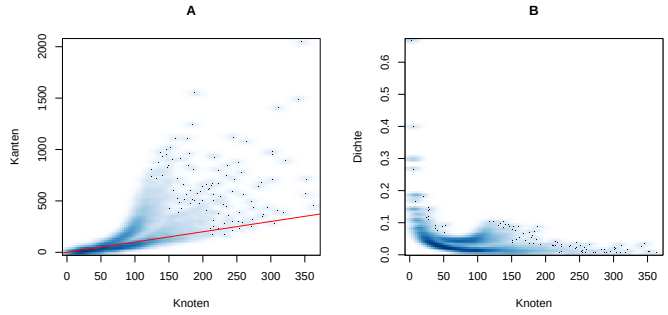

Abbildung 5.21: Für die Graphen aus Gruppe 2 ist die Anzahl der Knoten gegenüber der Anzahl der Kanten (A) beziehungsweise gegenüber der Dichte des Graphen (B) aufgetragen.

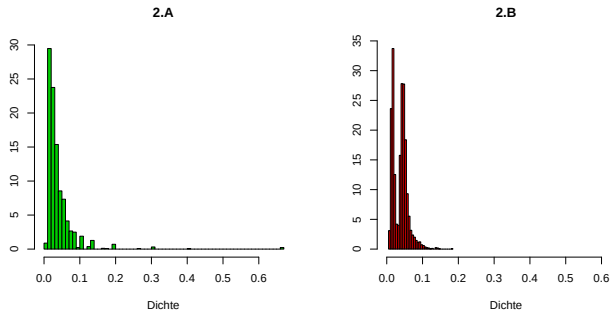

Abbildung 5.22: Für die prämoralisierbaren Graphen (2.A) und die nicht prämoralisierbaren Graphen (2.B) ist die Verteilung der Dichte dargestellt.

Graphen auftreten. Warum es diese zwei Kollektive gibt, ist nicht geklärt und kann ein Startpunkt für weitere Untersuchungen sein.

Zusammengefasst bestätigen sich die in Ansatz 1 bewiesenen Tatsachen und Vermutungen auch im Ansatz 2. Die Prämoralisierungseigenschaft hängt von der Anzahl der Knoten ab, und bei der Dichte lässt sich kein klarer Zusammenhang feststellen. Ein Test wie bei Ansatz 1 lässt sich bei Ansatz 2 aber nicht durchführen, denn hier gibt es keine disjunkten Genkollektive, da in jeder Genmenge die gleichen Klassifikations-Gene vorhanden sind.

5.3 Nutzen erzeugter Graphen zum Validieren von Algorithmen

Das Erzeugen von positiv definiten Matrizen mit Nebenbedingungen hat, wie in Kapitel 4 erläutert, eine Hauptanwendung bei der Validierung von Netzwerkalgorithmen, und deshalb wird in diesem Abschnitt der Algorithmus von Schäfer und Strimmer validiert, vor allem um beurteilen zu können, ob sich die Unterschiede in der Validierungsstrategie auf Spezifität und Sensitivität auswirken. Unterschiede ergeben sich in der von Schäfer und Strimmer vorgeschlagenen Validierung an zwei Stellen. Zum einen werden andere Graphen als vorgegebene Struktur gewählt. Diese sollen die biologischen Restriktionen besser repräsentieren als die bei Schäfer und Strimmer gewählten Graphen. Zum anderen soll zur Erstellung der positiv definiten Matrizen der in Kapitel 4 vorgestellte Ansatz benutzt werden. Es zeigt sich in dem folgenden Abschnitt, dass die Unterschiede einen starken Einfluss haben. Betrachtet man beispielsweise den *positive predictive value*, so schneidet hier der Algorithmus von Schäfer und Strimmer sehr viel schlechter ab als in der Validierungsstudie in [64].

Betrachtet man Matrizen, die mit Hilfe des Schäfer-Strimmer-Algorithmus aus den neun in Kapitel 5 vorgestellten Datensätzen in Ansatz 1 geschätzt worden sind, so sind die meisten Matrizen nicht diagonaldominant (85.5 %). Reduziert man diese Matrizen dann

auf die Einträge, die sich signifikant von Null unterscheiden (bei einer q-Wert-Schranke von 0.2), so sind noch 12.4 % nicht diagonaldominant. Das bedeutet, bei einem gewissen Anteil der erzeugten Matrizen liegt keine Diagonaldominanz vor. Der Validierungsansatz von Schäfer und Strimmer benutzt aber nur diagonaldominante Matrizen, so dass es sinnvoll ist, den Validierungsansatz auch auf nicht diagonaldominante Matrizen auszudehnen, wie dies mit dem PddP-Algorithmus möglich ist.

In ihren Arbeiten[64, 63] benutzen Schäfer und Strimmer einen einfachen Algorithmus, um zufällige Graphen zu erzeugen. Sie geben sich eine Knotenanzahl G und eine Kantenwahrscheinlichkeit η_A vor. Ein Graph mit G Knoten wird realisiert, indem für jede der $\binom{G}{2}$ möglichen Kanten ein Bernoulli-Experiment mit Erfolgswahrscheinlichkeit η_A durchgeführt wird. Eine Kante befindet sich dann in dem Graphen, falls das zugehörige Experiment erfolgreich gewesen ist. Legt man ein solches Modell zu Grunde, so ist die im letzten Abschnitt eingeführte Dichte eines Graphen ein Schätzer für η_A.

In [63] variieren Schäfer und Strimmer den Wert G in einem Bereich von $20 - 210$ und η_A in einem Bereich von 0.01 bis 0.2. In [64] wird G auf 100 gesetzt und η_A auf 0.04. Zu bemerken ist hier, dass der Algorithmus, welcher in dieser Studie untersucht wird und der von den Autoren als besser eingestuft wird, in [64] vorgestellt wird.

Die Anzahl X der Kanten eines Graphen, der wie oben geschrieben erzeugt wird, ist binomialverteilt zu den Parametern $\binom{G}{2}$ und η_A. Legt man die Werte aus [64] zu Grunde, so ergibt sich somit:

$$\begin{aligned} E(X) &= \binom{G}{2} \cdot \eta_A = 198 \\ SD(X) &= \sqrt{\binom{G}{2} \cdot \eta_A \cdot (1 - \eta_A)} = 13.79 \end{aligned}$$

Für eine vorgegebene Zahl $K \in \{0, \ldots \binom{G}{2}\}$ und eine vorgegebene Knotenanzahl G lässt sich ein η_A bestimmen, so dass die mittlere Kantenzahl der auf obige Weise erzeugten Graphen dem vorgegebenen K entspricht: $\eta_A = K \cdot \binom{G}{2}^{-1}$. Es ergibt sich dann für die Standardabweichung $SD(X) = \sqrt{K} \cdot \sqrt{1 - K \cdot \binom{G}{2}^{-1}}$. Das bedeutet, es ist mit diesem Ansatz möglich, sich für eine fest vorgeschriebene Anzahl von Kanten K ein η_A so zu wählen, dass die mittlere Anzahl von Kanten bei zufällig erzeugten Graphen K ist. Bei größerem K wird dann allerdings auch die Standardabweichung größer.

In Abbildung 5.23 wird für einen Knotenbereich zwischen 90 und 110 die Anzahl der Kanten der Graphen, die sich aus realen Microarray-Daten ergeben haben (siehe Kapitel 5.2) verglichen mit der Anzahl der Kanten, die sich durch Simulation aus dem Ansatz ergeben, der von Schäfer und Strimmer in [64] benutzt worden ist. Man erkennt, dass mit dem Ansatz der zufälligen Kantenverteilung vor allem Graphen mit mehr Kanten erzeugt werden als dies bei Graphen der Fall ist, die mit Hilfe des Algorithmus aus realen Microarray-Daten geschätzt worden sind. Dies wird auch durch die Abbildungen 5.14 und 5.21 verdeutlicht. Für Graphen mit vielen Knoten ist dort die Dichte, welche wie beschrieben einen Schätzer für η_A darstellt, zwar relativ konstant, allerdings ist der Wert viel

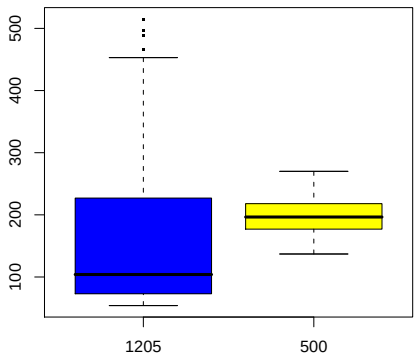

Abbildung 5.23: Für einen Knotenbereich von 90-110 ist die Anzahl der Kanten als Boxplot dargestellt. Hierbei wurden sowohl aus realen Microarray-Daten geschätzte Graphen (blau) als auch Graphen betrachtet, die mit dem Ansatz von Schäfer und Strimmer erzeugt wurden (gelb). Die Kantenwahrscheinlichkeit beträgt beim letzten Ansatz 0.04. Die Zahlen der x-Achse geben die Anzahl der Werte an, die in den Boxplot eingeflossen sind.

geringer als 0.04. Das bedeutet, dass die Strukturen, die bei Schäfer und Strimmer vorgegeben werden, nicht, oder nur zu einem Teil, die Strukturen realer Netze widerspiegeln, die aus Microarray-Daten geschätzt worden sind. Deshalb soll nun eine weitere Validierung des Schäfer-Strimmer-Algorithmus durgeführt werden. Als Grundlage für die Simulation werden die Graphen benutzt, die in Kapitel 5.2 geschätzt worden sind. Da die geschätzten Graphen auf Microarray-Daten basieren, ist damit die Hoffnung gegeben, dass sie die Struktur eines Gennetzwerkes gut widerspiegeln. Die Analyse der Graphen in Kapitel 5.2 hat ergeben, dass ein Großteil der Graphen prämoralisierbar ist, so dass all diese Graphen als Grundlage des in Kapitel 4 vorgestellten Simulationsansatzes benutzt werden können.

Abbildung 5.7 zeigt, dass der Schäfer-Strimmer-Algorithmus beim Schätzen von Netzwerken mit nur wenigen Knoten häufiger einen Fehler produziert. Aus diesem Grund werden für die folgenden Untersuchungen nur Graphen benutzt, die mindestens 30 Knoten haben. Diese Graphen werden nach der Größe der größten Zusammenhangskomponente sortiert und die ersten 150 werden für die Untersuchungen benutzt. Dies geschieht getrennt für die Ansätze 1 und 2 aus Kapitel 5.2. Zusammengenommen liegen also den folgenden Untersuchungen Graphen zu Grunde, die sich bei Ansatz 1 oder 2 in Gruppe 2.A befinden und deren Knotenanzahl mindestes 30 beträgt. Jeder einzelne Graph ist auf der Genebene und nicht auf der Spotebene erzeugt worden.

Für diese Untersuchung werden also 300 verschiedene Graphen betrachtet, von denen 24 nicht zerlegbar sind. Die Graphen haben zwischen 30 und 143 Knoten (Median: 55) und zwischen 16 und 111 Kanten (Median: 39). Vergleicht man diese Zahlen mit Abbildung 5.23, so erkennt man, dass die gewählten Graphen auch nur einen Teil der Graphenstrukturen

abdecken, die in realen Daten auftreten. Alle übrigen Graphen könnten aber als Grundlage für andere Simulationsansätze wie zum Beispiel diagonaldominante Matrizen oder Matrizen aus hyper-inversen Wishart-Verteilungen dienen.

Für jeden Graph wird eine Referenzmatrix erzeugt, die die Struktur des gewählten Graphen repräsentiert. Jeder Eintrag, der nicht auf Null festgelegt wird, wird mit einem zufälligen Wert zwischen -1 und -0.8 oder 0.8 und 1 besetzt. Die Diagonale der Matrix wird auf 1 gesetzt. Aus dieser Referenzmatrix werden mit Hilfe des PddP-Algorithmus 10 verschiedene partielle Korrelationsmatrizen beziehungsweise obere Dreiecksmatrizen erzeugt. Mit Hilfe dieser Matrizen werden Daten aus einer der partiellen Korrelationsmatrix zugehörigen multivariaten Normalverteilung generiert. Aus diesen Daten wird mit Hilfe des Algorithmus von Schäfer und Strimmer die partielle Korrelationsmatrix geschätzt und getestet, welche Einträge signifikant von Null abweichen. Um einen Graphen zu erzeugen, muss, wie in Kapitel 5.2, eine Schranke angegeben werden, bis zu welchem q-Wert ein Eintrag als signifikant angesehen werden soll und somit eine Kante in den Graphen gesetzt wird. Es werden hier drei Schranken für den q-Wert getrennt untersucht: 0.05, 0.1 und 0.2. Um den Einfluss der Anzahl der erzeugten Datenpunkte auf Sensitivität und Spezifität zu messen, wird der Algorithmus mit den Probenanzahlen 20, 30, 40, 50, 100, 150 und 200 durchgeführt, wobei die Daten mit einer kleineren Anzahl in den Daten mit einer größeren Anzahl enthalten sind.

5.3.1 Ergebnisse

Bei 19560 der 63000 erstellten Datensätze konnte keine Analyse durchgeführt werden, weil es entweder beim Schätzen oder Testen im Algorithmus von Schäfer und Strimmer einen Fehler gegeben hat. Von den übrigen 43440 Matrizen, die durch den PddP-Algorithmus erzeugt worden sind, sind alle nicht diagonaldominant. Somit wird der Algorithmus an einem anderen Kollektiv von Graphen getestet als in der Originalarbeit von Schäfer und Strimmer.

Die vom Schäfer-Strimmer-Algorithmus geschätzten Matrizen sind nach Konstruktion immer positiv definit. Aber nur 18 aller geschätzten Matrizen sind diagonaldominant, so dass ersichtlich ist, dass das Kollektiv, auf dem Schäfer und Strimmer ihren Algorithmus validiert haben, ein anderes ist als das, welches nun benutzt wird. Hier wird also eine andere Klasse von Matrizen abgedeckt, die aber bei Schätzungen aus Microarray-Daten durchaus auftreten.

Reduziert man die Matrizen auf die durch den Test als signifikant von Null abweichenden Einträge, so ist keine Matrix positiv definit. Dieses Ergebnis steht in Kontrast zu den vorherigen Ergebnissen. Dort wurde gezeigt, dass die aus realen Daten erzeugten Präzessionsmatrizen nach Reduktion auf die Elemente, die signifikant von Null abweichen, zu einem großen Teil positiv definit sind. Dies liegt aber vor allem daran, dass diese Matrizen diagonaldominant sind und die geschätzten partiellen Korrelationen in den Microarray-Daten klein sind. In diesem Ansatz werden zwar auch dünn besetzte Graphen vorgegeben, aber die einzelnen Korrelationen sind nahe bei 1 oder -1, so dass keine Diagonaldominanz vorliegt.

Die Ergebnisse sind in den Abbildungen 5.24, 5.25 und 5.26 dargestellt. Jede Abbildung repräsentiert die Ergebnisse für eine der drei q-Wert-Schranken. Diese gibt an, bis zu welchem q-Wert, der aus dem Test resultiert, ob ein Eintrag in der geschätzten Präzessionsmatrix als signifikant von Null verschieden angesehen werden kann, eine entsprechende Kanten in einen Graphen eingefügt wird.

Bei den Sensitivitäts- und Spezifitätsmessungen erkennt man, dass wie erwartet die Sensitivität mit steigender Probenanzahl ansteigt. Vergleicht man die Resultate mit denen aus [64], so fällt auf, dass in der dortigen Arbeit die Sensitivität viel geringer gewesen ist als in der aktuellen Studie. In [64] wird darauf hingewiesen, dass die schlechte Sensitivität darauf zurückzuführen ist, dass die dortigen geschätzten Matrizen nur kleine partielle Korrelationen haben. Die aktuelle Studie bestätigt nun diese These. Die hier benutzten partiellen Korrelationen sind groß und aus diesem Grund ist auch die Sensitivität besser. Ein weiterer Unterschied in der Simultaionsstrategie sind die benutzten Graphen. Wie gezeigt, wurden bei Schäfer und Strimmer nur Graphen mit 100 Knoten und einer vorher festgesetzten durchschnittlichen Kantenzahl betrachtet, während in dieser Studie Graphen mit unterschiedlichsten Knotenanzahlen und Dichten zu Grunde gelegt wurden. Es wurden vor allem Graphen benutzt, die wenige Kanten haben und das scheint für den Algorithmus vorteilhaft zu sein.

Es ist sinnvoll, bei den Auswertungen der Daten auch den *positive predictive value* (ppv) und nicht nur die Spezifität zu betrachten. Dadurch, dass alle Graphen dünn besetzt sind, ist die Zahl der nicht vorhandenen Kanten sehr groß, häufig im Bereich von 3000-4000. Aus diesem Grund wird selbst eine relativ große Anzahl von falsch positiven Kanten die Spezifität nicht stark reduzieren. Bei dem *positive predictive value* geht die Anzahl aller vorhandenen Kanten ein und die Anzahl aller gefundenen Kanten, so dass sich ein besseres Bild ergibt und falsch gefundene Kanten ein größeres Gewicht bekommen. Wie man in den Abbildungen erkennen kann, ist der *positive predictive value* gering. Man findet also mit den Algorithmen immer viel mehr Kanten als wirklich vorhanden sind. Dieses Ergebnis steht im Widerspruch zu den Ergebnissen aus [64]. Dort ist der *positiv predictive value* immer bei ca. 0.9, das bedeutet, fast alle gefundenen Kanten waren auch richtige Kanten. In [64] liegt die Anzahl der gefundenen Kanten bei ungefähr 120, falls viele simulierte Proben zu Grunde liegen. In dieser Studie ist die Anzahl der gefundenen Kanten vor allem bei einer kleinen Probenzahl sehr viel größer.

Abbildung 5.27 zeigt die Verteilung der absoluten Korrelationswerte der gefundenen und nicht gefundenen Kanten für eine q-Wert-Schranke von 0.1 aufgeteilt nach der Anzahl der zur Verfügung stehenden Beobachtungen. Man erkennt, dass sich keine Unterschiede bezüglich der Probenanzahl ergeben. Zudem ist ersichtlich, dass der Schäfer-Strimmer-Algorithmus vor allem Kanten mit sehr großer partieller Korrelation findet, was nicht verwunderlich ist. Allerdings werden auch viele Kanten nicht gefunden, die eine hohe partielle Korrelation haben. Die Verteilungen für die anderen q-Wert Schranken sind sehr ähnlich zu den Ergebnissen bei einer Schranke von 0.1 und hier nicht dargestellt.

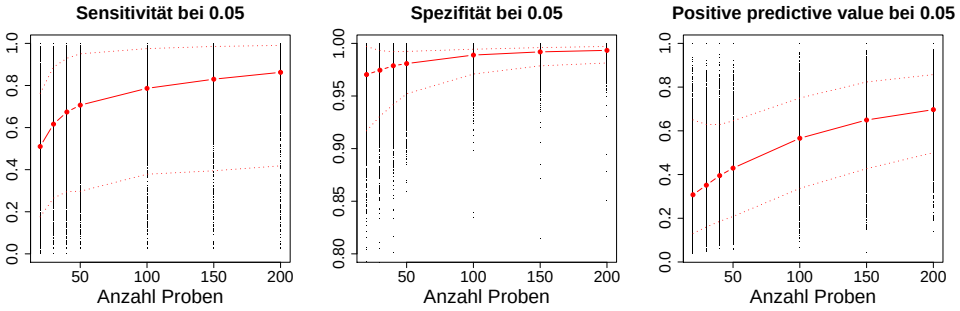

Abbildung 5.24: Sensitivität, Spezifität und ppv für verschiedene Anzahlen von eingegangenen Proben. Die rote Linie zeigt den Median an und die gepunktete Linie das 10- beziehungsweise 90-Prozent-Quantil. Die q-Wert-Schranke ist auf 0.05 gesetzt.

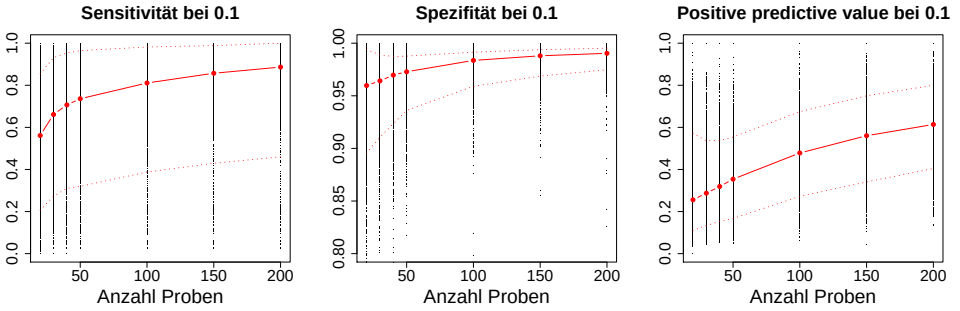

Abbildung 5.25: Sensitivität, Spezifität und ppv für verschiedene Anzahlen von eingegangenen Proben. Die rote Linie zeigt den Median an und die gepunktete Linie das 10- beziehungsweise 90-Prozent-Quantil. Die q-Wert-Schranke ist auf 0.1 gesetzt.

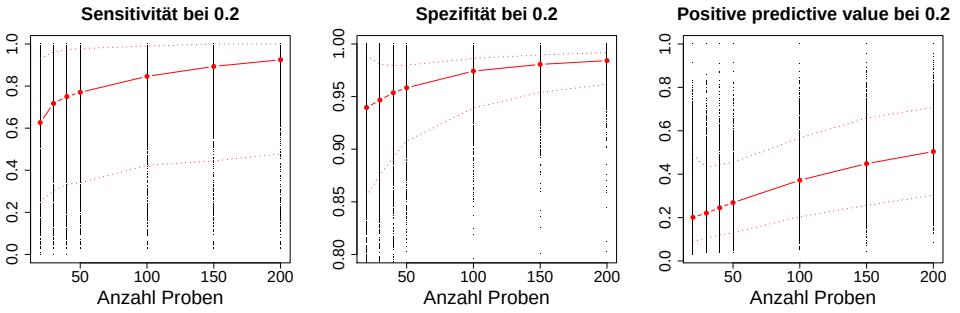

Abbildung 5.26: Sensitivität, Spezifität und ppv für verschiedene Anzahlen von eingegangenen Proben. Die rote Linie zeigt den Median an und die gepunktete Linie das 10- beziehungsweise 90-Prozent-Quantil. Die q-Wert-Schranke ist auf 0.2 gesetzt.

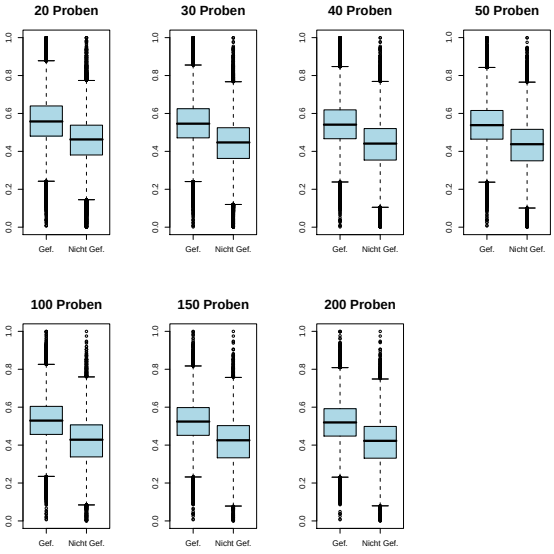

Abbildung 5.27: Die absoluten Werte der Korrelation für die gefundenen und nicht gefundenen Kanten. Die q-Wert-Schranke ist auf 0.1 gesetzt.

5.4 Ähnlichkeit von Graphstrukturen

Obwohl der Begriff Systembiologie und die häufig damit verbundene Schätzung von Netzwerken und Graphen ein sehr aktuelles Thema ist und sich viele Studien damit befassen, Interaktionen zwischen Genen zu finden, ist Vergleichbarkeit von Graphen ein Punkt, welchem in diesem Bereich noch keine große Bedeutung zugekommen ist. Nur sehr wenige Arbeiten, beispielsweise eine Arbeit von Balasubramanian[2], befassen sich damit, aus Microarray-Daten geschätzte Strukturen zu vergleichen und so Rückschlüsse über Gemeinsamkeiten oder Unterschiede zwischen Graphen darzustellen. In diesem Abschnitt wird nun ein weiterer Ansatz vorgestellt, der mit Hilfe des parametrischen Bootstrap Strukturen vergleicht. Dieser Ansatz wird auf zwei verschiedene Fragestellungen angewandt.

Bei der ersten Frage interessiert man sich dafür, ob es bei KEGG-Pathways zwischen definierten Probenuntergruppen bei den erzeugten graphischen Strukturen mehr als zufällige Unterschiede gibt. Hierfür wird der Datensatz von Hannenhalli [33] und der Datensatz von Wang [75] benutzt. Es ergibt sich, dass bei dem vorgestellten Ansatz einige Pathways als signifikant gefunden werden, wohingegen durch den Ansatz von Balasubramanian[2] kein signifikantes Resultat gefunden wird.

Bei der zweiten Frage untersucht man, ob es zwischen Genen aus einem Pathway und Genen, die aus einem Klassifikator stammen, signifikant mehr Kanten gibt als durch Zufall. Die Knotenmenge eines Graphen besteht somit aus den Genen eines Pathways (Pathway-Gene) zusammen mit den Genen, die durch einen Klassifikationsalgorithmus gefunden worden sind und für die Klassifikation benutzt werden (Klassifikations-Gene). Für diese

Betrachtungen wird der Datensatz von Wang [75] benutzt. Es ergibt sich hier, dass bei fast allen Pathways signifikante Zusammenhänge existieren. Bevor die zwei Ansätze im Detail betrachtet werden, wird zunächst der Permutationsansatz von Balasubramanian vorgestellt.

5.4.1 Permutationsansatz von Balasubramanian

Wie beschrieben wird für die erste Fragestellung der neu vorgestellte Ansatz mit Hilfe des parametrischen Bootstrap verglichen mit dem von Balasubramanian[2] vorgestellten Ansatz über Kanten- oder Knotenpermutationen. Das Ziel bei diesem Ansatz besteht darin, zu testen, ob es mehr Gemeinsamkeiten zwischen zwei gegebenen Graphen gibt als bei zufällig erzeugten Graphen. Hierzu betrachtet man die Graphen $H_1 = (V, E_1)$ und $H_2 = (V, E_2)$. Beiden Graphen liegt die gleiche Knotenmenge V mit $n = |V|$ zu Grunde. Die zu untersuchende Statistik ist definiert durch

$$S(H_1, H_2) := \#(E_1 \cap E_2),$$

das bedeutet, die Statistik berechnet die Anzahl der Kanten, die sich sowohl in H_1 als auch in H_2 befinden. Um die Signifikanz einer solchen Statistik zu bestimmen, werden zwei mögliche Verfahren vorgeschlagen:

- **Permutation der Kanten:** Die Kanten des Graphen H_2 werden permutiert, das bedeutet, es entsteht ein zufälliger Graph $\tilde{H}_2 = (V, \tilde{E}_2)$ mit der Bedingung $\#\tilde{E}_2 = \#E_2$. Dann berechnet man $X_i := S(H_1, \tilde{H}_2)$.

- **Permutation der Knoten:** Die Nummerierung des Graphen H_2 wird permutiert, das bedeutet, man erzeugt eine zufällige Permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ und wendet diese auf die vorhandene Nummerierung von H_2 an, um eine neue Nummerierung der Knoten zu bekommen. Es entsteht ein Graph $\tilde{H}_2 = (V, \tilde{E}_2)$. Dann berechnet man $X_i := S(H_1, \tilde{H}_2)$.

Sowohl bei der Knotenpermutation als auch bei der Kantenpermutation wird der Vorgang i mal wiederholt. Der p-Wert ist definiert als der Anteil der Durchgänge, bei denen $S(H_1, \tilde{H}_2)$ mindestens so groß ist wie $S(H_1, H_2)$, das bedeutet:

$$p := \frac{\#\{X_i | X_i \geq S(H_1, H_2)\}}{i}.$$

Zu bemerken ist noch, dass das Ziel der folgenden Untersuchung darin besteht, Unterschiede zwischen Graphen zu finden und keine Gemeinsamkeiten. Dafür muss aber nur die oben definierte Statistik S abgeändert werden. Hierauf wird auch im folgenden Abschnitt eingegangen.

5.4.2 Frage 1: Unterschiede auf Patientengruppen

Um Unterschiede auf Patientengruppen zu untersuchen, betrachtet man einen Datensatz, in dem zwei Patienten-Untergruppen auf Grund eines vorher bestimmten klinischen Parameters vertreten sind. In dieser Arbeit wird der Datensatz von Hannenhalli[33] und der Datensatz von Wang[75] benutzt. Diese beiden Datensätze sind die größten der neun in Kapitel 5 vorgestellten Datensätze. Zudem werden in beiden Datensätzen zwei biologisch sehr unterschiedliche Phämomene untersucht. Im Datensatz von Hannenhalli [33] werden die Untergruppen dilatative Kardiomyopathie(DCM) und ischämische Kardiomyopathie(ICM) gebildet. Wie schon in Kapitel 1 beschrieben, wird davon ausgegangen, dass bei univariaten Untersuchungen nicht viele differentielle Gene zwischen diesen Gruppen gefunden werden. Bei den Untergruppen im Wang-Datensatz handelt es sich um den viel untersuchten Vergleich zwischen Estrogen-Rezeptor-positiven und -negativen Patientenproben. Hier findet man auf transkriptioneller Ebene schon bei univariaten Untersuchungen sehr viele Unterschiede zwischen diesen zwei Gruppen [65]. Interessant ist es also zu beobachten, wie der vorgestellte Algorithmus bei unterschiedlichen Voraussetzungen abschneidet.

Wie schon in Kapitel 5.2 erläutert, ist es nicht sinnvoll, Netzwerke auf dem kompletten Satz vorhandener Gene zu erzeugen. Somit sollte auch bei dem Vergleich von Graphenstrukturen eine Reduktion vorgenommen werden, und dies ist auch im Sinn der Fragestellung, denn es interessiert häufig mehr, ob sich Strukturen in einem kleinen definierten Bereich ändern oder nicht. Es werden hier die Reduktionen zu Grunde gelegt, die auch schon bei Ansatz 1 in Kapitel 5.2 beschrieben worden sind. Es wird untersucht, ob sich graphische Strukturen bei vorher definierten KEGG-Pathways unterscheiden, wobei wieder die Pathways benutzt werden, bei denen sich mehr als 30 Gene im Pathway befinden.

Der parametrische Bootstrap für diese Untersuchung ist in Abbildung 5.28 skizziert. Für jeden Datensatz und jeden Pathway wird zuerst aus den Gesamtdaten ein GGM geschätzt. Hierzu wird der Algorithmus von Schäfer und Strimmer[64] benutzt, um die normierte Präzessionsmatrix A zu schätzen. Zusätzlich wird zu einem Niveau von 0.2 getestet, ob sich die Einträge signifikant von Null unterscheiden, wobei das Niveau gewählt wird wie in [64]. Die Elemente aus A, die sich nicht signifikant von Null unterscheiden, werden dann auf Null gesetzt. Es entsteht die Matrix A'. Diese ist nun aber nicht notwendigerweise positiv definit, wie die Untersuchungen in Kapitel 5.3 gezeigt haben. Falls die Matrix nicht positiv definit ist, wird in einem Zwischenschritt der PddP-Algorithmus angewandt, um zu der geschätzten Matrix eine positiv definite Matrix zu finden, die bezüglich der Frobeniusnorm einen möglichst geringen Abstand zu A' hat. Die resultierende Matrix wird mit $\bar{\Omega}$ bezeichnet. Ist die Matrix A' schon positiv definit, so gilt $\bar{\Omega} = A'$. Mit Hilfe der aus dem Schäfer-Strimmer-Algorithmus geschätzten partiellen Varianzen erhält man aus $\bar{\Omega}$ die Präzessionsmatrix Ω. Die Verteilung $N(0, \Omega^{-1})$ fungiert dann als Nullverteilung der Gene eines Pathways für den parametrischen Bootstrap. Die Nullhypothese dieses Ansatzes ist also, dass die Präzessionsmatrizen in den beiden Patienten-Untergruppen gleich sind und somit die Daten der Untergruppen aus dem gleichen GGM erzeugt worden sind.

Für die weiteren Schritte sei n_1 die Anzahl der Proben in Untergruppe 1 und n_2 die der Untergruppe 2. Für beide Untergruppen werden ebenfalls Matrizen Ω_1' und Ω_2' wie

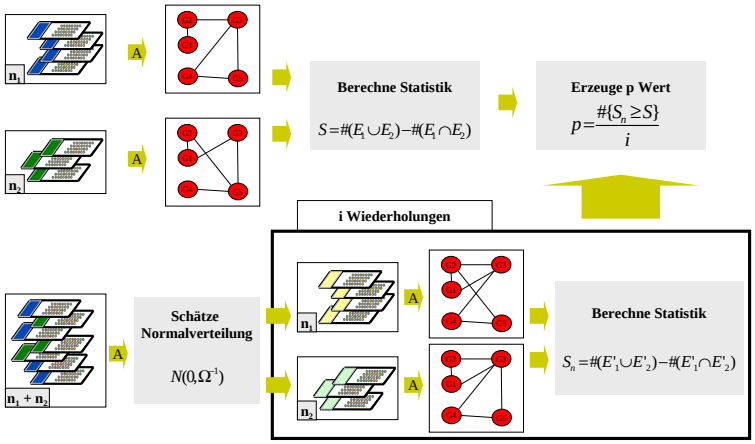

Abbildung 5.28: Eine Skizze für den parametrischen Bootstrap zur Beantwortung von Frage 1. Die Farben der Microarrays repräsentieren die verschiedenen Untergruppen. Ein mit einem A gekennzeichneter Pfeil gibt an, dass hier der Algorithmus von Schäfer und Strimmer benutzt worden ist.

oben beschrieben erzeugt und als ungerichtete Graphen $H_1 = (V, E_1)$ und $H_2 = (V, E_2)$ repräsentiert. Es wird dann folgende Statistik $S(H_1, H_2)$ berechnet, die die Unterschiede zwischen den Graphen misst:

$$S(H_1, H_2) := \#((E_1 \cup E_2) \backslash (E_1 \cap E_2)).$$

Diese Statistik zählt die Anzahl der Kanten, die sich entweder nur in H_1 oder nur in H_2 befinden, aber nicht in beiden Graphen zusammen. Falls sich die Graphen H_1 und H_2 stark in der Besetzung der Kanten unterscheiden, wird diese Statistik einen großen Wert annehmen, während Graphen, bei denen sich nur wenige Kanten unterscheiden, eine kleine Statistik haben.

Um die Signifikanz der Statistik zu berechnen, werden Realisierungen der Verteilung $N(0, \Omega^{-1})$ generiert, wobei hierfür das in Kapitel 2 beschriebene Verfahren benutzt wird. Es werden zuerst n_1 Realisierungen erzeugt, und von diesen wird mit Hilfe des Algorithmus von Schäfer und Strimmer eine Matrix geschätzt und daraus der Graph B_1 generiert. Zum anderen werden n_2 Realisierungen erzeugt, aus denen in gleicher Art und Weise der Graph B_2 generiert wird. Man berechnet $X_i = S(B_1, B_2)$. Der gesamte Vorgang wird i Mal wiederholt und man definiert dann den p-Wert als

$$p := \frac{\#\{X_i | X_i \geq S(H_1, H_2)\}}{i}.$$

Der Ansatz von Balasubramanian[2] hat ein anderes Ziel als der oben beschriebene Ansatz. Während sich beim parametrischen Bootstrap die Statistik aus der Zahl der Kanten berechnet, die sich in der Vereinigung der beiden Graphen, aber nicht im Schnitt befindet, so wird bei Balasubramanian als Statistik die Anzahl der gemeinsamen Kanten gewählt. Der Grund hierfür ist, dass bei Balasubramanian getestet wird, ob die zwei untersuchten Graphen viele Gemeinsamkeiten haben, während in dem parametrischen Bootstrap untersucht wird, ob die zwei Graphen viele Unterschiede haben. Modifiziert man nun aber die beiden Graphen, die bei Balasubramanian als Eingabe genommen werden, so kann man die gleiche Statistik erhalten. Hierzu definiert man für einen beliebigen ungerichteten Graphen $H = (V, E)$ die Menge

$$E^c := \{\{v, w\} | v, w \in V \wedge v \neq w \wedge \{v, w\} \notin E\}.$$

Mit dieser Definition gilt für die im parametrischen Bootstrap benutzte Statistik $S(H_1, H_2)$:

$$S(H_1, H_2) = \#((E_1 \cap E_2^c) \cup (E_1^c \cap E_2)).$$

Für den Permutationsansatz von Balasubramanian werden die Graphen $H_1 = (V, E_1)$ und $H_2 = (V, E_2)$ nun wie folgt modifiziert. Man definiert

$$\begin{aligned} E_3 &:= E_1 \cup E_2 \\ E_4 &:= E_1^c \cup E_2^c \\ H_3 &:= (V, E_3) \\ H_4 &:= (V, E_4). \end{aligned}$$

Benutzt man nun für den Permutationstest die Graphen H_3 und H_4, so ergibt sich für die Statistik

$$\begin{aligned} E_3 \cap E_4 &= (E_1 \cup E_2) \cap (E_1^c \cup E_2^c) \\ &= (E_1 \cap E_1^c) \cup (E_1 \cap E_2^c) \cup (E_2 \cap E_1^c) \cup (E_2 \cap E_2^c) \\ &= (E_1 \cap E_2^c) \cup (E_2 \cap E_1^c). \end{aligned}$$

Somit entspricht die im Permutationsansatz berechnete Statistik der Statistik des Ansatzes über den parametrischen Bootstrap und die resultierenden p-Werte sind vergleichbar.

Da man in diesen Untersuchungen viele Hypothesen gleichzeitig testet, werden die resultierenden p-Werte – beim parametrischen Bootstrap und bei den Permutationstests – gegen multiples Testen korrigiert. Hier werden die Werte durch den Ansatz von Benjamini und Yekutieli [5] adjustiert, um die *false discovery rate* zu kontrollieren.

Ergebnisse

Es gibt 86 Pathways mit mehr als 30 Genen. Von diesen konnten im Brustkrebsdatensatz nur 83 untersucht werden, da bei 3 Pathways kein parametrischer Bootstrap durchgeführt werden konnte, weil sich entweder der Graph der gemeinsamen Verteilung oder der Graph

der einzelnen Verteilungen nicht schätzen ließ. Bei dem Herzdatensatz konnten alle Pathways untersucht werden. Erstaunlich ist die Tatsache, dass alle erzeugten Präzessionsmatrizen in beiden Studien positiv definit gewesen sind, so dass der PddP-Algorithmus hier keine Anwendung findet. Betrachtet man die Ergebnisse genauer, so lässt sich dies teilweise begründen: der Großteil der geschätzten reduzierten partiellen Korrelationsmatrizen ist schon diagonaldominant (86 % in Brustkrebsdatensatz und 87 % im Herzdatensatz), denn die Anzahl der Kanten in der gemeinsamen Verteilung ist gering (Mittelwert ca. 27.12).

Die Ergebnisse der Untersuchung sind in Tabelle 5.8 für den Brustkrebsdatensatz gezeigt und in Tabelle 5.9 für den Herzdatensatz. Sowohl bei der Kantenpermutation als auch bei der Knotenpermuation ist kein Pathway signifikant. Dies ist nicht verwunderlich. Bei beiden Permutationsansätzen wird die Signifikanz dadurch getestet, dass die Unterschiede eines Graphen A zu einem anderen Graphen B signifikant sind, falls diese sehr häufig größer sind als Unterschiede zwischen dem Graphen A und einem zufälligen Graphen, wobei an diesen zufälligen Graphen einige wenige Bedingungen geknüpft sind (Anzahl der Kanten wie Graph B oder die komplette Kantenstruktur wie Graph B). Nun sind Graphen, die aus Microarray-Daten geschätzt werden, dünn besetzt, also *sparse*. Man hat also wenig Kanten in dem zufälligen Graphen, aber viele Möglichkeiten, diese Kanten zu platzieren. Insbesondere gibt es viele Möglichkeiten, diese Kanten so zu platzieren, dass es viele Unterschiede zu dem vorgegebenen Graphen A gibt. Die Wahrscheinlichkeit, dass bei einer zufälligen Kantenbesetzung höchstens eine gewisse Anzahl von Kanten den Kanten aus A entsprechen, ist in solchen Situationen sehr groß und lässt sich mit Hilfe der hypergeometrischen Verteilung berechnen (siehe auch [2]). Diese Berechnungen bestätigen, dass es im Fall der Kantenpermutation bei den gegebenen Voraussetzungen nie mehr Unterschiede gibt als bei einer zufälligen Besetzung.

Bei dem parametrischen Bootstrap werden andere Voraussetzungen gemacht als bei dem Kanten- oder Knotenpermutationstest. Es wird davon ausgegangen, dass die Präzessionsmatrizen der beiden definierten Gruppen identisch sind und somit können feinere Strukturunterschiede besser gefunden werden. Im Brustkrebsdatensatz gibt es 12 Pathways mit einem adjustierten p-Wert von unter 0.05 und im Herzdatensatz gibt es 20 Pathways. Das ist insofern ein überraschendes Ergebnis, als dass bei unvariaten Analysen sehr viel mehr signifikante Gene im Brustkrebsdatensatz zu verzeichnen sind. Die erzeugten Graphen einiger signifikanter Pathways befinden sind im Anhang dieser Arbeit. Bei genauerer Betrachtung der Resultate fällt auf, dass von den signifikanten Pathways des Brustkrebsdatensatzes nur ein Pathway nicht auch signifikant im Herzdatensatz ist (Abbildung 5.29, A). Betrachtet man die korrigierten p-Werte aufgetragen gegen die Knotenzahl des untersuchten Pathways, so ist hier eine klare Tendenz zu erkennen (Abbildung 5.29, B und C). Die Pathways mit sehr vielen Knoten sind in beiden Pathways signifikant. Bei solchen Graphen stehen mehr Variablen zur Verfügung, an denen sich Unterschiede festmachen lassen, und somit ist die Power größer. Aber es ist auch von der biologischen Seite her sinnvoll, dass einige dieser Pathways signifikant unterschiedlich sind.

Im Brustkrebsdatensatz gibt es signifikante Pathways oder Genfamilien, die schon mit Brustkrebs und dem Estrogen-Rezeptor-Status in Verbindung gebracht worden sind. Man

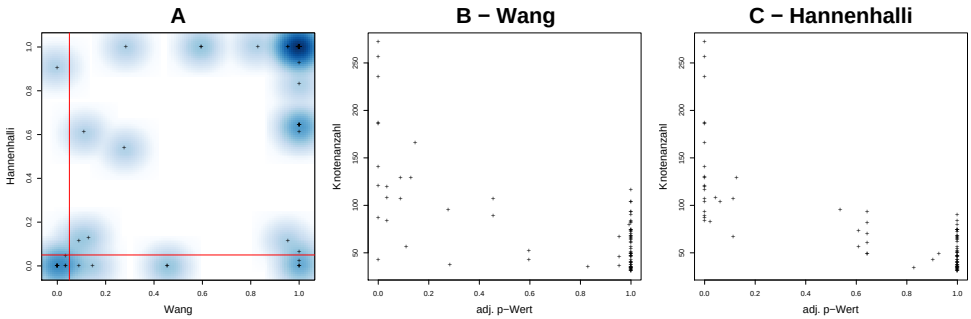

Abbildung 5.29: Für die Resultate des parametrischen Bootstraps sind die adjustierten p-Werte der Pathways aufgetragen, die in beiden Datensätzen auftreten (A). Zudem sind für den Wang Datensatz (B) und den Hannenhalli Datensatz (C) die korrigierten p-Werte gegen die Knotenzahl aufgetragen.

weiß, dass der Estrogen-Rezeptor-Status mit der Immuninfiltration korreliert ist [68]. Dies korreliert mit dem Resultat, dass die ersten signifikanten Pathways der Liste (*Cell adhesion molecules*, *Cytokine-cytokine receptor interaction*,*Hematopoietic cell linage* und *Focal adhesion*) mit immunregulatorischen Genen assoziiert sind. Betrachtet man weitere signifikante Pathways, so wirkt auf den ersten Blick der *Type I diabetes mellitus* Pathway deplatziert. Doch in diesem Pathway befinden sich viele HLA Gene, die zu der Klasse der MHC Antigene gehören und es wurde in Brustkrebs-Zelllinien gezeigt, dass *MHC class 1 Antigene* mit der Expression des Estrogen-Rezeptor-Status korreliert sind [54]. Wie oben beschrieben, gibt es Zusammenhänge zwischen dem Estrogen-Rezeptor-Status und der Immuninfiltration, und diese ist wiederum korreliert mit der Expression der MHC Antigene. Daher ist es plausibel, dass es beim Estrogen-Rezeptor-Status signifikante Unterschiede in der Interaktion zwischen den HLA Genen gibt.

Neben der Immunreaktion spielt auch die Proliferation eine zentrale Rolle bei Brustkrebs. Estrogen-Rezeptor-negative Tumore sind mit einer schlechteren Prognose [68] assoziiert. Aus diesem Grund ist es nicht verwunderlich, dass in dieser Analyse auch Pathways gefunden wurden, die bei der Kontrolle der Zellproliferation eine Rolle spielen, wie beispielsweise der *MAPK signaling* Pathway [53] und der *JAK-STAT signaling* Pathway [74]. Für *JAK-STAT* ist in Expressionsanalysen gezeigt worden, dass dieser mit dem Estrogen-Rezeptor-Status assoziert ist [78]. Zudem ist in [15] eine 'MAPK signatur' entwickelt worden, die zwischen den Estrogen-Rezeptor-positiven und -negativen Tumoren unterscheiden kann.

Betrachtet man die Ergebnisse im Herzdatensatz, so fällt auf, dass viel mehr Interaktionen in der Untergruppe der ICM gefunden worden sind. Eine mögliche Erklärung ist, dass es sich hier im Vergleich zu der DCM Gruppe um eine homogenere Gruppe handelt. Bei den ICM Fällen ist die Ursache der Erkrankung, eine Verengung der das Herz versorgenden Koronargefäße, bekannt, in der DCM Gruppe werden alle übrigen Fälle zusammen gefasst, deren Ursache unbekannt sind [52]. Unter den signifikanten Pathways im

Hannenhalli Datensatz gibt es viele, die mit dem Vergleich von DCM und ICM beziehungsweise allgemein mit Herzerkrankungen in Verbindung gebracht werden. Die Interaktion der extrazellulären Matrix (ECM) mit verschiedenen Zellkomponenten, aber auch die Kommunikation zwischen den glatten Muskelzellen spielt im Herz bei der Morphologie und bei der Kontraktilität eine wichtige Rolle [31]. Es wurde gezeigt, dass sich verschiedene extrazelluläre Matrixproteine wie Fibronektin zwischen DCM und ICM unterscheiden [34]. Weitere wichtige Gene für die ECM-Interaktion und die Zell-Zell-Kommunikation sind Laminine und Kollagene, die Interaktionen in den Pathways bilden. Der *Hematopoietic cell linage* Pathway sowie der *Toll-like receptor signaling* Pathway können mit Entzündungsprozessen und Artheriosclerose assoziiert werden. Es ist zudem bekannt, dass chronische Entzündungen und virale Infektionen eine mögliche Ursache für Herzerkrankungen bilden. In einem Anteil von Patienten mit DCM ist zudem enterovirale RNA gefunden worden [61] und die Expression des Toll-like-Rezeptor 4 ist in diesen Patienten mit dem Anteil an enteroviraler RNA korreliert [60]. Zudem sind viele Toll-like-Rezeptoren (TLR2, TLR3, TLR4, TLR5, TLR7 and TLR9) in Herzgeweben exprimiert [8]. Die *Oxidative Phosphorylierung* ist ein biochemischer Prozess zur Gewinnung von Energie in Form von ATP. Im Herzen ist die Energiegewinnung beispielsweise wichtig für die Kontraktilität, so dass Veränderungen in der Oxidativen Phosphorylierung auch zu Herzerkrankungen führen können [48, 25, 38].

Insgesamt zeigen die Untersuchungen sowohl des Herzdatensatzes als auch des Brustkrebsdatensatzes, dass es bei einigen der gefundenen Pathways von der biologischen Seite gute Argumente gibt, dass diese sich signifikant zwischen den Untergruppen unterscheiden. Gerade beim Herzdatensatz, bei dem es in der univariaten Analyse nur sehr wenige signifikante Gene gibt, können nun die neuen Hypothesen, die sich aus den Daten ergeben, beispielsweise der Einfluss des *Toll-like receptor signaling* Pathways, genauer untersucht werden.

5.4.3 Frage 2: Zusammenhänge zwischen Pathway-Genen und Klassifikations-Genen

Möchte man Unterschiede zwischen zwei Patientengruppen untersuchen und benutzt dazu eine univariate Analyse, so findet diese Analyse alle Gene, die zwischen diesen Gruppen differentiell sind. Mit diesen Resultaten ist es dann möglich, weitere Analysen durchzuführen. Naheliegend ist die Berechnung, ob Gene, die gewissen vorher definierten Gruppen zugeordnet sind, in den univariaten Analysen signifikant überrepräsentiert sind. Eine ähnliche Strategie möchte man auch bei Genen anwenden, die bei einer Klassifikation gefunden werden. Als Beispiel sei hier wieder der PAM Algorithmus[71] genannt. Dieser Algorithmus nimmt eine Variablenselektion mit Hilfe eines Schrumpfungsansatzes vor, so dass die Anfangs große Menge von Genen auf wenige reduziert wird. In dem Wang Datensatz, der für die Frage 2 untersucht wird, reduziert sich die Menge auf 133 Klassifikations-Gene (siehe Tabelle 5.6). Benutzt man nun aber diese Gene, um zu testen, ob beispielsweise ein Pathway überrepräsentiert ist, so ergibt sich ein Problem. Während das Ziel einer univariaten Analyse darin besteht, alle Gene zu finden, die zwischen zwei Gruppen differentiell

	Name	V	S(G1,G2)	E1	E2	p0	p1	p2
1	Cell adhesi..Ms)	120	276	261	41	0.000	1	1
2	Cytokine-cy..ion	235	175	111	74	0.000	1	1
3	Hematopoiet..age	87	170	169	19	0.000	1	1
4	Focal adhes	186	125	73	60	0.000	1	1
5	Type I diab..tus	42	122	112	48	0.000	1	1
6	Jak-STAT si..way	140	29	0	29	0.000	1	1
7	Regulation ..ton	187	28	5	23	0.000	1	1
8	MAPK signal..way	256	11	3	10	0.000	1	1
9	Neuroactive..ion	272	8	0	8	0.000	1	1
10	Leukocyte t..ion	108	89	60	39	0.035	1	1
11	ECM-recepto..ion	83	82	63	29	0.035	1	1
12	Natural kil..ity	119	27	23	10	0.035	1	1
13	Tight junct	107	28	16	18	0.089	1	1
14	Wnt signali..way	129	19	0	19	0.089	1	1
15	Metabolism ..450	56	85	79	32	0.111	1	1
16	Insulin sig..way	129	13	2	11	0.130	1	1
17	Calcium sig..way	165	6	0	6	0.147	1	1
18	Gap junctio	95	38	34	18	0.277	1	1
19	Pyruvate me..ism	37	90	84	20	0.284	1	1
20	Cell Commun..ion	106	61	39	36	0.455	1	1
21	Toll-like r..way	89	59	8	59	0.455	1	1
22	Androgen an..ism	42	47	44	13	0.596	1	1
23	Tyrosine me..ism	52	35	2	35	0.596	1	1
24	Glutathione..ism	35	49	39	30	0.830	1	1
25	ABC transpo..ral	36	51	11	54	0.953	1	1
26	Lysine degr..ion	45	24	24	0	0.953	1	1
27	Glycerophos..ism	66	13	6	9	0.953	1	1
28	TGF-beta si..way	79	22	13	11	0.993	1	1

Tabelle 5.8: Ergebnisse der Pathwayanalyse bei Wang. $E1/2$ ist die Anzahl der signifikanten Kanten in der ER-/ER+ Untergruppe. $S(G1, G2)$ ist die Anzahl der unterschiedlichen Kanten und V die Anzahl der Knoten in einem Pathway. $p0$ ist der adjustierte p-Wert für den parametrischen Bootstrap, $p1$ der adjustierte p-Wert für die Kantenpermutation und $p2$ der adjustierte p-Wert für die Knotenpermutation. Die Liste ist aufsteigend nach $p0$ geordnet, wobei alle Pathways mit $p0$ kleiner 1 dargestellt sind. Für gleiche Werte $p0$ wird absteigend nach $S(G1, G2)$ sortiert.

	Name	V	S(G1,G2)	E1	E2	p0	p1	p2
1	MAPK signal..way	256	402	87	329	0.000	1	1
2	Focal adhes	186	333	121	238	0.000	1	1
3	Regulation ..ton	187	254	82	184	0.000	1	1
4	Cytokine-cy..ion	235	209	109	110	0.000	1	1
5	Neuroactive..ion	272	204	112	104	0.000	1	1
6	ECM-recepto..ion	83	183	39	164	0.000	1	1
7	Cell adhesi..Ms)	120	122	74	68	0.000	1	1
8	Calcium sig..way	165	90	22	74	0.000	1	1
9	Hematopoiet..age	87	83	11	82	0.000	1	1
10	Oxidative p..ion	103	78	4	78	0.000	1	1
11	Jak-STAT si..way	140	75	46	37	0.000	1	1
12	Toll-like r..way	89	61	9	60	0.000	1	1
13	Cell Commun..ion	106	57	32	31	0.000	1	1
14	Ribosome	93	43	36	17	0.000	1	1
15	Purine meta..ism	130	29	0	29	0.000	1	1
16	Natural kil..ity	119	18	6	12	0.000	1	1
17	Axon guidan	116	15	1	14	0.000	1	1
18	Wnt signali..way	129	14	4	14	0.000	1	1
19	Antigen pro..ion	82	90	44	74	0.023	1	1
20	Leukocyte t..ion	108	18	0	18	0.043	1	1
21	Cell cycle	103	22	10	22	0.062	1	1
22	Glycerophos..ism	66	35	34	1	0.113	1	1
23	Tight junct	107	6	3	3	0.113	1	1
24	Insulin sig..way	129	9	6	5	0.126	1	1
25	Gap junctio	95	10	6	10	0.537	1	1
26	Metabolism ..450	56	35	23	32	0.610	1	1
27	Long-term d..ion	73	5	5	0	0.610	1	1
28	Pathogenic ..HEC	49	54	19	51	0.643	1	1
29	Pathogenic ..PEC	49	54	19	51	0.643	1	1
30	B cell rece..way	60	17	0	17	0.643	1	1
31	Apoptosis	81	11	0	11	0.643	1	1
32	Tryptophan ..ism	70	10	10	2	0.643	1	1
33	GnRH signal..way	93	4	0	4	0.643	1	1
34	Bile acid b..sis	34	41	10	43	0.828	1	1
35	Type I diab..tus	42	66	0	66	0.904	1	1
36	Valine, leu..ion	49	29	29	2	0.927	1	1

Tabelle 5.9: Ergebnisse der Pathwayanalyse bei Hannenhalli. $E1/2$ ist die Anzahl der signifikanten Kanten in der DCM/ICM Untergruppe. $S(G1, G2)$ ist die Anzahl der unterschiedlichen Kanten und V die Anzahl der Knoten in einem Pathway. $p0$ ist der adjustierte p-Wert für den parametrischen Bootstrap, $p1$ der adjustierte p-Wert für die Kantenpermutation und $p2$ der adjustierte p-Wert für die Knotenpermutation. Die Liste ist aufsteigend nach $p0$ geordnet, wobei alle Pathways mit $p0$ kleiner 1 dargestellt sind. Für gleiche Werte $p0$ wird absteigend nach $S(G1, G2)$ sortiert.

expremiert sind, ist ein Klassifikationsalgorithmus daran interessiert, die Anzahl der Parameter möglichst stark zu reduzieren. Dies sei an einem Beispiel erläutert: Es seien Gen 1 und Gen 2 gegeben, die beide zwischen zwei Patientenkollektiven differentiell sind. Gen 1, welches zum Pathway A gehört, sei stark mit Gen 2, welches nicht zum Pathway A gehört, korreliert. Ein Klassifikationsalgorithmus möchte die Informationen nutzen, die Gen 1 und Gen 2 besitzen. Da diese Informationen aber identisch sind und das Ziel eines Klassifikationsalgorithmus auch darin besteht, möglichst wenig Gene zu nutzen, wird nur eines der beiden Gene zufällig ausgewählt. Ist dies Gen 1, so ist der Pathway A überrepräsentiert, ist es Gen 2, so ist dies nicht der Fall. Betrachtet man nun aber ein Netzwerk basierend auf Korrelation oder partieller Korrelation, wobei die Knotenmenge aus der Vereinigung der Klassifikations-Gene und der Gene des zu untersuchenden Pathways bestehen, so wird es in dem oberen Beispiel eine Kante zwischen Gen 1 und Gen 2 geben. Selbst wenn dann Gen 1 nicht also Klassifikations-Gen gewählt worden ist, gibt es so einen Hinweis darauf, dass es Assoziationen zwischen den Pathway- und den Klassifikations-Genen gibt. Ein Problem ergibt sich, falls einige der Pathway-Gene auch Klassifikations-Gene sind. In diesem ersten Ansatz werden deshalb nur die Pathways untersucht, für die das nicht der Fall ist. Sind Pathway-Gene auch Klassifikations-Gene, so gibt es auf jeden Fall Assoziationen.

Mit dem folgenden Test soll untersucht werden, ob man die Nullhypothese verwerfen kann, dass es keine Verbindungen zwischen Pathway-Genen und Klassifikations-Genen gibt. Somit ist die Zielsetzung hier eine andere als im ersten Ansatz, wodurch auch die zu nutzende Statistik und das Vorgehen geändert werden muss. Man ist hier interessiert an der Anzahl der Pathway-Gene, die mindestens eine Kante zu einem Klassifikations-Gen besitzen, und möchte vergleichen, ob diese signifikant überrepräsentiert sind. Es kommt also hier darauf an, die mit Klassifikations-Genen verbundenen Pathway-Gene zu zählen. Diese werden dann verglichen mit Werten bei Graphen, die aus Daten entstehen, bei denen man weiß, dass es keine Interaktion gibt. Es werden in diesem Ansatz also auch die Strukturen von Graphen verglichen, allerdings entstehen diese Graphen dadurch, dass man Knotenmengen einmal gemeinsam betrachtet und einmal getrennt. Im Detail werden durch den parametrischen Bootstrap Daten aus einer Verteilung erzeugt, bei der es zwischen Klassifikations-Genen und Pathway-Genen keine partiellen Korrelationen gibt. Für die Klassifikations-Gene und die Pathway-Gene werden getrennt, wie in Kapitel 5.4.2 beschrieben, Präzessionsmatrizen $\Omega^K = (\omega_{ij}^K) \in M(n_K, n_K, \mathbb{R})$ und $\Omega^P = (\omega_{ij}^P) \in M(n_P, n_P, \mathbb{R})$ aus den vorliegenden Microarray-Daten erzeugt. Die Anzahl der Klassifikations-Gene ist hierbei n_K und die Anzahl der Pathway-Gene ist n_P. Diese beiden Blockmatrizen werden zu der Matrix $\Omega = (\omega_{ij}) \in M(n_K + n_P, n_K + n_P, \mathbb{R})$ zusammen gefügt, mit

$$\begin{aligned}
\omega_{ij} &= \omega_{ij}^P \quad \text{für } i \leq n_P \text{ und } j \leq n_P \\
\omega_{ij} &= 0 \quad \text{für } i \leq n_P \text{ und } j > n_P \\
\omega_{ij} &= 0 \quad \text{für } i > n_P \text{ und } j \leq n_P \\
\omega_{ij} &= \omega_{ij}^K \quad \text{für } i > n_P \text{ und } j > n_P
\end{aligned}$$

Die Matrix ist positiv definit und die zugehörige Verteilung $N(0, \Omega^{-1})$ ist die angenommene

Nullverteilung. Für jeden einzelnen Pathway wird, wie in 5.4.2 beschrieben, ein ungerichteter Graph $G = (V, E)$ erzeugt, wobei die Knoten so geordnet werden, dass die ersten n_P Knoten die Pathway-Gene repräsentieren. Die Statistik $S(G)$ ist dann definiert als

$$S(G) := \#\{v \in V | v \in \{v_1, \ldots, v_{n_P}\} \wedge \text{ es existiert ein } w \in \{v_{n_P+1}, \ldots, v_n\} \text{ mit } \{v, w\} \in E\}.$$

Wie im vorherigen Abschnitt werden zur Berechnung des p-Wertes Daten aus $N(0, \Omega^{-1})$ erzeugt, um die Signifikanz zu berechnen. Das komplette Vorgehen wird in diesem Fall getrennt für die einzelnen Untergruppen des Wang-Datensatzes, also die Gruppe der Estrogen-Rezeptor-positiven und -negativen Proben, durchgeführt. Die aus den parametrischen Bootstraps erzeugten p-Werte werden wieder durch Benjamini und Yekutieli [5] adjustiert, um die *false discovery rate* zu kontrollieren.

Ergebnisse

Genau wie in Kapitel 5.4.2 wurden nur Pathways mit mindestens 30 Genen untersucht. Von den untersuchten 86 Pathways wurden 43 nicht benutzt, da hier Klassifikations-Gene und Pathway-Gene teilweise übereingestimmt haben. Von den übrigen 86 Vergleichen konnten 3 nicht ausgewertet werden, da es hier, wie bei Fragestellung 1, auch Fehler bei den Berechnungen gegeben hat. Die Ergebnisse des parametrischen Bootstraps sind in Tabelle 5.10, 5.11 und 5.12 aufgeführt. Neben dem Namen des Pathways und dem Namen der Untergruppe sind die Anzahl der Pathway-Gene (es gibt immer 133 Klassifikations-Gene), die Anzahl der Pathway-Gene mit einer Kante zu Klassifikations-Genen sowie die Ergebnisse des parametrischen Bootstraps dargestellt. Erstaunlicherweise sind fast alle Pathways signifikant, selbst wenn die Anzahl der Pathway-Gene mit einer Kante zu Klassifikations-Genen gering ist. Somit gibt es bei fast allen Vergleichen einen signifikanten Zusammenhang zwischen den Klassifikations-Genen und einem Pathway. Der Mehrwert dieser Aussage ist für den vorliegenden Vergleich somit gering. Interessanterweise ist aber zu erkennen, dass es vor allem Verbindungen zwischen den Klassifikations-Genen und den Pathway-Genen in der Untergruppe der ER- Patienten gibt. Dieses Ergebnis kann als Startpunkt weiterer Untersuchungen genutzt werden.

	Name	Gruppe	PW	S(G1,G2)	p0
1	Jak-STAT si..way	ER-	140	126	0.0000
2	Purine meta..ism	ER-	130	111	0.0000
3	Tight junct	ER-	107	78	0.0000
4	Apoptosis	ER-	81	71	0.0000
5	Hematopoiet..age	ER-	87	70	0.0000
6	Colorectal ..cer	ER-	72	69	0.0000
7	Pyrimidine ..ism	ER-	74	65	0.0000
8	VEGF signal..way	ER-	68	64	0.0000
9	Adherens ju..ion	ER-	74	63	0.0000
10	Hematopoiet..age	ER+	87	63	0.0000
11	Antigen pro..ion	ER-	82	60	0.0000
12	Antigen pro..ion	ER+	82	57	0.0000
13	Cell cycle	ER-	103	57	0.0000
14	Glycan stru..s 2	ER-	54	54	0.0000
15	Glycolysis ..sis	ER-	56	54	0.0000
16	Glycerophos..ism	ER-	66	52	0.0000
17	Adipocytoki..way	ER-	66	52	0.0000
18	Inositol ph..ism	ER-	47	47	0.0000
19	Purine meta..ism	ER+	130	47	0.0000
20	Ribosome	ER-	93	45	0.0000
21	Glycerolipi..ism	ER-	51	43	0.0000
22	Ubiquitin m..sis	ER-	40	38	0.0000
23	Type II dia..tus	ER-	42	38	0.0000
24	Pathogenic ..HEC	ER-	49	38	0.0000
25	Pathogenic ..PEC	ER-	49	38	0.0000
26	Oxidative p..ion	ER+	103	38	0.0000
27	Oxidative p..ion	ER-	103	37	0.0000
28	ABC transpo..ral	ER-	36	36	0.0000
29	Taste trans..ion	ER-	35	35	0.0000
30	Notch signa..way	ER-	39	35	0.0000

Tabelle 5.10: Ergebnisse des Vergleiches zwischen Klassifikations-Genen und Pathway-Genen im Wang-Datensatz. $S(G1, G2)$ bezeichnet die Anzahl der Pathway-Gene mit einer Kante zu Klassifikations-Genen und $p0$ ist der gegen multiples Testen adjustierte p-Wert. Die Liste ist aufsteigend nach $p0$ geordnet. Für gleiche Werte $p0$ wird absteigend nach $S(G1, G2)$ sortiert.

	Name	Gruppe	PW	S(G1,G2)	p0
31	mTOR signal..way	ER-	47	35	0.0000
32	Bile acid b..sis	ER-	34	34	0.0000
33	N-Glycan bi..sis	ER-	35	34	0.0000
34	Linoleic ac..ism	ER-	31	31	0.0000
35	SNARE inter..ort	ER-	32	31	0.0000
36	Glycerophos..ism	ER+	66	31	0.0000
37	Sphingolipi..ism	ER-	31	30	0.0000
38	Basal trans..ors	ER-	32	30	0.0000
39	Neurodegene..ers	ER-	34	30	0.0000
40	Fructose an..ism	ER-	37	30	0.0000
41	Apoptosis	ER+	81	30	0.0000
42	Ribosome	ER+	93	30	0.0000
43	ATP synthes	ER-	34	29	0.0000
44	Type I diab..tus	ER-	42	29	0.0000
45	Glycan stru..s 2	ER+	54	29	0.0000
46	Cell cycle	ER+	103	28	0.0000
47	Tight junct	ER+	107	27	0.0000
48	Folate bios..sis	ER-	33	25	0.0000
49	Glycolysis ..sis	ER+	56	25	0.0000
50	Proteasome	ER-	31	24	0.0000
51	SNARE inter..ort	ER+	32	24	0.0000
52	Pyruvate me..ism	ER-	37	24	0.0000
53	Adipocytoki..way	ER+	66	24	0.0000
54	Colorectal ..cer	ER+	72	24	0.0000
55	Cholera - I..ion	ER-	37	22	0.0000
56	Jak-STAT si..way	ER+	140	21	0.0000
57	Nicotinate ..ism	ER-	34	20	0.0000
58	ATP synthes	ER+	34	20	0.0000
59	ABC transpo..ral	ER+	36	19	0.0000
60	Glycerolipi..ism	ER+	51	19	0.0000

Tabelle 5.11: Ergebnisse Klassifikations-Gene Wang (Fortsetzung 1)

	Name	Gruppe	PW	S(G1,G2)	p0
61	Adherens ju..ion	ER+	74	18	0.0000
62	Cholera - I..ion	ER+	37	16	0.0000
63	Fructose an..ism	ER+	37	15	0.0000
64	Neurodegene..ers	ER+	34	14	0.0000
65	Nicotinate ..ism	ER+	34	14	0.0000
66	Bile acid b..sis	ER+	34	13	0.0000
67	Ubiquitin m..sis	ER+	40	16	0.0060
68	mTOR signal..way	ER+	47	15	0.0060
69	VEGF signal..way	ER+	68	15	0.0060
70	Linoleic ac..ism	ER+	31	10	0.0119
71	N-Glycan bi..sis	ER+	35	11	0.0175
72	Notch signa..way	ER+	39	12	0.0288
73	Type II dia..tus	ER+	42	10	0.0449
74	Inositol ph..ism	ER+	47	10	0.0449
75	Pyruvate me..ism	ER+	37	11	0.0498
76	Folate bios..sis	ER+	33	9	0.0546
77	Sphingolipi..ism	ER+	31	8	0.1132
78	Taste trans..ion	ER+	35	8	0.1224
79	Proteasome	ER+	31	10	0.1314
80	Pathogenic ..PEC	ER+	49	13	0.1557
81	Pathogenic ..HEC	ER+	49	13	0.1671
82	Basal trans..ors	ER+	32	8	0.1671
83	Type I diab..tus	ER+	42	13	0.3401

Tabelle 5.12: Ergebnisse Klassifikations-Gene Wang (Fortsetzung 2)

Kapitel 6

Diskussion und Ausblick

Die Umkehrung des Moralisierungsvorganges eines gerichteten Graphen ist in der Literatur bis jetzt nur an wenigen Stellen, beispielsweise in einer Arbeit von Madigan[47], erwähnt worden. Obwohl der in dieser Arbeit als *Prämoralisierung* definierte Vorgang vor allem als technisches Hilfsmittel genutzt worden ist, um positiv definite Matrizen mit Nebenbedingungen zu erzeugen, sind in dieser Arbeit auch grundlegende Eigenschaften der prämoralisierbaren Graphen untersucht worden. Beispielsweise wurde gezeigt, dass eine Prämoralisierung, falls sie existiert, nicht eindeutig ist, was aber für den Algorithmus zur Erstellung von positiv definiten Matrizen mit Nebenbedingungen zuerst kein Nachteil ist, da man nur eine beliebige Prämoralisierung benötigt. Aus diesem Grund ist die Anzahl aller Prämoralisierungen für einen gegebenen Graphen nicht näher untersucht worden. Ein möglicher Ansatzpunkt zur Beantwortung dieser Frage liegt darin, Funktionen zu definieren, die von einer Prämoralisierung eines Graphen zu einer weiteren Prämoralisierung wechseln, beispielsweise durch Hinzufügen oder Drehen einer gerichteten Kante. So könnte man den Raum der Prämoralisierungen eines Graphen durchwandern, um so weitere Informationen über dessen Struktur zu erhalten. Im Hinblick auf den Optimierungsansatz aus Kapitel 4 wäre es beispielsweise von Interesse, ob es Prämoralisierungen eines Graphen H gibt, die sich besser für die Optimierung der Funktion OP_A eignen als andere. Man kann sich in diesem Zusammenhang fragen, ob es sogar für eine vorgegebene Matrix A eine optimale Prämoralisierung gibt, der Raum der möglichen Prämoralisierungen also ein Minimum oder Maximum bezüglich der Resultate aus OP_A aufweist.

Neben einigen notwendigen Bedingungen ist in Kapitel 3 gezeigt worden, dass die Menge der prämoralisierbaren Graphen die Menge der zerlegbaren Graphen als eine echte Teilmenge enthält. Zerlegbarkeit ist somit eine hinreichende aber nicht notwendige Eigenschaft für Prämoralisierung, und für zerlegbare Graphen ist es sehr einfach, eine Prämoralisierung zu finden. Von Interesse sind aber nicht nur die zerlegbaren Graphen, sondern auch die prämoralisierbaren, aber nicht zerlegbaren Graphen. Somit besteht ein Großteil von Kapitel 3 aus dem Algorithmus, der für einen beliebigen ungerichteten Graphen eine Prämoralisierung findet, falls eine solche existiert. Dieser Algorithmus stellt das einzige notwendige und hinreichende Kriterium für die Prämoralisierungseigenschaft dar. Hier wäre ein einfacheres und leichter zu berechnendes Kriterium wünschenswert, falls man an einer konkreten

Prämoralisierung nicht interessiert ist, denn obwohl in dieser Arbeit keine Komplexitätsberechnungen gemacht worden sind, hat sich gezeigt, dass für große Graphen die Berechnung im ungünstigsten Fall sehr lange dauern kann. Hierfür verantwortlich sind die Schritte 6-8, in denen Teilmengen $M' \subseteq M$ der Möglichkeitenmenge M aus dem Graphen entfernt werden müssen. Die Anzahl der möglichen Mengen M' steigt mit der Größe der Menge M, die wiederum mit der Anzahl der Knoten korreliert ist. Erste Ansatzpunkte zur Verbesserung der Rechenzeit des Algorithmus liefern die Schritte 2 und 6. In Schritt 2 des Algorithmus wird die Menge M reduziert, was die Anzahl der möglichen Teilmengen verringert. Es wird eine Kante $\{v, w\}$ aus M und dem Graphen H entfernt, falls diese nicht mehr zu einer Moralisierung beitragen kann. Das ist der Fall, falls v und w keinen gemeinsamen Nachbarn haben oder aber alle gemeinsamen Nachbarn durch Kanten mit v und w verbunden sind, die in der Möglichkeitenmenge liegen. Hier wären zusätzliche Kriterien wünschenswert, die weitere Kanten aus M entfernen und so die Geschwindigkeit des Algorithmus weiter verbessern.

Wie groß die Menge der prämoralisierbaren Graphen ist, ist in dieser Arbeit nur empirisch für graphische Strukturen untersucht worden, die aus Microarray-Daten erzeugt worden sind. Für diese Klasse von Graphen hängt der Anteil der prämoralisierbaren Graphen gegenläufig von der Anzahl der Knoten ab, das bedeutet, vor allem Graphen mit wenigen Knoten sind prämoralisierbar und der Anteil der prämoralisierbaren Graphen sinkt mit steigender Knotenanzahl. Die Dichte eines Graphen, das heißt der Anteil der gesetzten Kanten an allen möglichen Kanten, scheint keinen Einfluss darauf zu haben, ob ein Graph prämoralisierbar ist oder nicht (siehe Kapitel 5.2). Eine Begründung für dieses Verhalten ist noch nicht gefunden worden, wobei für die Abhängigkeit von der Knotengröße möglicherweise Schwellenfunktionen bei Zufallsgraphen (siehe [22]) einen ersten Hinweis liefern. Hierzu nimmt man an, dass die aus Microarray-Studien erzeugten Graphen durch Zufallsgraphen mit Wahrscheinlichkeit $p > 0$ beschrieben werden können, wobei p unabhängig von n ist. Die Größe n ist hierbei die Anzahl der Knoten in den untersuchten Graphen, und p ist die Wahrscheinlichkeit, mit der eine Kante gesetzt wird. Die Unabhängigkeitsannahme wird für große n durch die Abbildungen 5.14 und 5.21 gestützt. Betrachtet man unter diesen Voraussetzungen erzeugte Graphen H für ansteigende Knotenzahl n, so kann man beweisen, dass $\lim_{n\to\infty} P(H \text{ hat einen Zykel}) = 1$ gilt, das bedeutet für Graphen mit vielen Knoten gibt es mit großer Wahrscheinlichkeit Kreise. Dies ist aber die notwendige Bedingung dafür, dass ein Graph nicht prämoralisierbar ist. Obwohl in der Arbeit gezeigt worden ist, dass man nur durch Nutzung von Zufallsgraphen mit festem p nicht alle Graphen abdeckt, die aus Microarray-Daten geschätzt werden, ergibt sich hier doch ein möglicher Startpunkt für weitere Untersuchungen, um vielleicht Schwellenfunktionen dafür zu finden, dass ein Graph prämoralisierbar ist.

Betrachtet man einen nicht prämoralisierbaren Graphen H, so kann man durch Hinzufügen von Kanten immer erreichen, dass der Graph zerlegbar und somit prämoralisierbar ist. Somit ist es von Interesse zu erfahren, wie viele Kanten, im Durchschnitt und möglicherweise abhängig von der Anzahl der Knoten, man zu einem nicht prämoralisierbaren Graphen hinzufügen oder entfernen muss, damit dieser prämoralisierbar wird. Wenn diese Anzahl gering ist, so bestünde die Möglichkeit, für Simulationsstudien bei Microarray-Daten die nicht

prämoralisierbaren Graphen *leicht* zu verändern, und somit den Algorithmus A anwenden zu können. Damit würde die Menge der Graphen, die man für eine Simulationsstudie nutzen kann, vergrößert.

Daten aus Microarray-Analysen haben naturgemäß sehr viel mehr Variablen als Beobachtungen, so dass viele Methoden zur Erstellung von Netzwerken, die für Daten aus anderen Bereichen anwendbar sind, hier keine Verwendung finden. Ist man beispielsweise an der Korrelation zweier Variablen gegeben alle übrigen Variablen interessiert, um so einen Graphen zu erzeugen, so können diese Werte aus der normierten Präzessionsmatrix abgelesen werden [40]. Allerdings ist das Schätzen dieser Matrix nicht problemlos möglich, wenn mehr Variablen als Beobachtungen vorliegen, denn in einem solchen Fall ist die *sample Matrix*, der empirische Schätzer für die Kovarianzmatrix, mit Wahrscheinlichkeit 1 nicht von vollem Rang, also nicht invertiertbar [19]. Deshalb muss man bei Microarray-Daten zu anderen Verfahren greifen, um die Präzessionsmatrix und somit die partiellen Korrelationen zu erhalten.

Betrachtet man die geschätzten partiellen Korrelationen und testet, welche dieser partiellen Korrelationen Null sind, so steht dieses Vorgehen in engem Zusammenhang mit dem Lernen von Strukturen bei Gaußschen graphischen Modellen(GGM). Ist ein Gaußsches graphisches Modell zu einem ungerichteten Graphen $H = (V, E)$ gegeben, so liegt eine multivariate Normalverteilung vor, wobei die Einträge der Präzessionsmatrix und so auch der partiellen Korrelationsmatrix Null sind, bei denen im Graphen an der entsprechenden Stelle keine Kante vorliegt. Für die Elemente der Präzessionsmatrix, die zu einer Kante korrespondieren, sind keine Bedingungen gegeben. Deshalb ist es auch nicht sinnvoll zu fragen, welcher Graph dem GGM zu Grunde liegen kann, denn dies ist beispielsweise immer für den vollständigen Graphen der Fall. Statt dessen möchte man einen Graphen mit möglichst wenig Kanten finden, der dem GGM zu Grunde liegt. Verfahren, die testen, ob sich Einträge der Präzessionsmatrix signifikant von Null unterscheiden, geben Aussagen darüber, dass zwischen einigen Variablen ein Zusammenhang existiert, dass also gewisse Kanten in dem zu suchenden Graphen vorhanden sein müssen, und helfen somit, den minimalen Graphen zu finden, der dem GGM zu Grunde liegt.

In der letzten Zeit sind viele Artikel erschienen, in denen Netzwerke aus Microarray-Daten mit Hilfe von der Theorie der Gaußschen graphischen Modelle geschätzt werden (beispielsweise [63, 64, 50, 18, 12]). Die in den Arbeiten vorgestellten Algorithmen versuchen mit verschiedenen Methoden, mit den Problemen umzugehen, die durch die Struktur der Microarray-Daten vorgegeben sind. Dazu bedienen sich diese Algorithmen häufig heuristischer Elemente, deren Verhalten selten, und dann vor allem für den asymptotischen Fall, nachvollzogen werden kann. Selbst wenn asymptotisch korrektes Verhalten bewiesen worden ist, wie in [50], bleibt für den Anwender die Frage, wie sich der Algorithmus für die Probengrößen verhält, die in dem aktuellen Experiment vorliegen. Beispielsweise liefert der Algorithmus von Schäfer und Strimmer erst bei einer Gruppengröße von ca. 50 Beobachtungen gute Ergebnisse bezüglich Sensitivität und Spezifität [64]. Das bedeutet, für kleinere Gruppengrößen ist der Algorithmus nicht gut geeignet. Simulationsstudien sind also auch für den Anwender ein wichtiges Hilfsmittel, um sicher zu stellen, dass der vor-

geschlagene Algorithmus für die Gruppengröße funktioniert, die zur Verfügung steht. Das Ziel dieser Arbeit liegt auch darin, Hilfen bei Fragen zu liefern, die mit der Durchführung einer Simulationsstudie verbunden sind.

Betrachtet man eine Simulationsstudie wie in Abbildung 4.5, so ist der Ausgangspunkt einer solchen Studie ein ungerichteter Graph. Doch welche Strukturen beziehungsweise welche ungerichteten Graphen bilden einen sinnvollen Ausgangspunkt für die Studie? Benutzt man für die Validierung Strukturen, die bei Gen-Gen-Interaktionen nicht auftreten, so wird der Algorithmus auf einem Kollektiv validiert, welches die tatsächlichen Datenstrukturen nicht repräsentiert. Da bis jetzt nur wenige gesicherte Erkenntnisse über Gen-Gen-Interaktionen bekannt sind, und somit keine gesicherten Graphen vorliegen, die als Ausgangspunkt genommen werden können, besteht ein naheliegender Ansatz darin, für eine Simulationsstudie Graphen auszuwählen, die vorher aus Microarray-Daten geschätzt worden sind. Wenn ein netzwerkerzeugender Algorithmus die komplette zu Grunde liegende Struktur findet, so sind die erzeugten Graphen eine gute Ausgangsbasis für Simulationsstudien. Dies ist mit Sicherheit eine sehr optimistische Vorstellung, doch die Hoffung besteht, dass Graphen, die aus Microarray-Daten geschätzt worden sind, näher an den wirklichen Strukturen liegen als beispielsweise die Zufallsgraphen, die in der Arbeit von Schäfer und Strimmer benutzt worden sind (siehe Abbildung 5.23). Auf Grund der beschriebenen Idee wurden in dieser Arbeit als Ausgangspunkt der Simulationsstudie Graphen gewählt, die mit Hilfe des Algorithmus von Schäfer und Strimmer in verschiedenen Datensätzen bei verschiedenen q-Wert-Schranken für verschiedene Pathways erzeugt worden sind. Da der beschriebene Simulationsansatz benutzt worden ist, um später den Schäfer-Strimmer-Algorithmus zu validieren, könnte dies zu Problemen führen, falls der Schäfer-Strimmer-Algorithmus komplett falsche Strukturen erzeugt. In einem solchen Fall hat man keinen Unterschied zu anderen vorgegebenen Graphen wie den oben beschriebenen Zufallsgraphen. Um dieses Risiko zu minimieren, sollte für die Zukunft das beschriebene Vorgehen insofern erweitert werden, dass nicht nur Graphen, die in verschiedenenen Netzwerken unter verschiedenen Bedingungen getestet worden sind, in die Validierung einfließen, sondern auch solche, die von anderen Algorithmen erzeugt worden sind. Dass die Wahl der zu Grunde liegenden Strukturen und auch die Art der Matrixerzeugung (siehe nächsten Abschnitt) einen großen Einfluss auf die Spezifitäts- und Sensitivitätsberechnungen haben können, zeigt Kapitel 5.3 dieser Arbeit, in dem das beschriebene Simulationsverfahren benutzt wurde, um den Algorithmus von Schäfer und Strimmer zu validieren. Die Ergebnisse waren im Vergleich zu der ursprünglichen Publikation besser bezüglich der Sensivität, aber viel schlechter was den *positive predictive value* betrifft.

Ein weiterer wichtiger Punkt einer Simulationsstudie betrifft die Kovarianzmatrix beziehungsweise die Präzessionsmatrix, somit beschäftigt sich auch ein großer Teil der Arbeit mit der Bereitstellung von positiv definiten Matrizen mit Nebenbedingungen. Der in dieser Arbeit vorgestellte PddP-Algorithmus benutzt die beschriebene Prämoralisierung eines Graphen, um solche Matrizen zu erstellen. Der Algorithmus bietet Vorteile gegenüber anderen gängigen Verfahren, zeigt aber auch Nachteile. Zur Verdeutlichung sei hier zuerst einmal der PddP-Algorithmus mit den diagonaldominanten Matrizen verglichen, das ein-

fachste Verfahren zur Erzeugung positiv definiter Mattrizen mit Nebenbedingungen.

Mit diagonaldominanten Matrizen ist es für einen beliebigen ungerichteten Graphen möglich, eine positiv definite Matrix zu erzeugen, die die Struktur des Graphen repräsentiert. Hier liegt der Vorteil gegenüber dem PddP-Algorithmus, welcher voraussetzt, dass der gegebene Graph prämoralisierbar ist. Somit ist der PddP-Algorithmus nicht für jeden beliebigen Graphen anwendbar, es ist in dieser Arbeit aber gezeigt worden (Kapitel 5.2), dass viele der Graphen, die aus Microarray-Daten geschätzt worden sind, prämoralisierbar sind. Obwohl man bei diagonaldominanten Matrizen beliebige graphische Strukturen vorgeben kann, ist es bei Graphen, bei denen Gene viele Nachbarn haben, nicht möglich, alle Einträge mit großen Korrelationen zu besetzen (siehe Kapitel 4.3), was aber für die Auswertung einer Simulationsstudie von Vorteil sein kann. Mit dem PddP-Algorithmus hat man nicht so strenge Restriktion wie bei den diagonaldominanten Matrizen. Die Simulationsstudien in Kapitel 4 haben gezeigt, dass es durch den PddP-Algorithmus möglich ist, die freien Einträge einer Matrix mit hohen Korrelationen zu besetzen. Somit werden insbesondere Matrizen erstellt, die nicht diagonaldominant sind.

Der in Kapitel 2 beschriebene Ansatz von Roverato[56] zur Erstellung von positiv definiten Matrizen mit Nebenbedingungen zeigt viele Ähnlichkeiten zu dem PddP-Algorithmus, aber auch Unterschiede. Die Gemeinsamkeit liegt darin, dass in beiden Ansätzen eine obere Dreiecksmatrix K so mit Werten besetzt wird, dass für die Matrix $\Omega = K^t \cdot K$ gilt, dass $\Omega \in SPN(H, n)$. Somit liefern beide Algorithmen auch eine Zerlegung der erzeugten Matrix Ω und erleichtern somit die Erzeugung von Daten aus der zugehörigen multivariaten Normalverteilung. In der Art der Besetzung der oberen Dreiecksmatrix unterscheiden sich der Roverato-Algorithmus und der PddP-Algorithmus. Beim PddP-Algorithmus werden Elemente der Dreiecksmatrix K als Null vorgegeben. Diese Vorgaben beruhen auf der Prämoralisierung G für einen vorgegebenen ungerichteten Graphen H. Für jede beliebige Besetzung der übrigen Elemente gilt $\Omega \in SPN(H, n)$. Im Ansatz von Roverato werden zuerst einige Elemente der Matrix K beliebig vorgegeben. Die Elemente, die vorgegeben werden, korrespondieren zu den vorhandenen Kanten im ungerichteten Graphen H. Im Artikel von Roverato wird gezeigt, dass die übrigen Elemente der oberen Dreiecksmatrix so besetzt werden können, dass die finale Matrix Ω die Bedingung $\Omega \in SPN(H, n)$ erfüllt. Nach der Besetzung der beliebigen Elemente der Dreiecksmatrix K müssen also danach die übrigen Elemente der Dreiecksmatrix noch berechnet werden. Diese zusätzliche Berechnung ist beim PddP-Algorithmus nicht nötig.

Es ist nun, sowohl im Ansatz von Roverato als auch im PddP-Algorithmus, möglich, die frei wählbaren Elemente so zu optimieren, dass die resultierende Matrix bezüglich einer vorgegebenen Norm möglichst nahe an einer vorgegegebenen Matrix A liegt. Hier ergibt sich ein Vorteil für den PddP-Algorithmus, denn im Ansatz von Roverato müssen wie beschrieben für jede Besetzung der vorgegebenen Elemente die übrigen Elemente der oberen Dreiecksmatrix noch berechnet werden, während im PddP-Algorithmus die nicht freien Elemente auf Null gesetzt sind, was bei Optimierungsansätzen Zeit spart, sobald eine Prämoralisierung ermittelt worden ist. Andererseits hat man beim Ansatz von Roverato mehr frei zu besetzende Variablen, denn dort kann für jede Kante ein Wert an die entsprechenden Stelle in der Matrix gesetzt werden, während im PddP-Ansatz nur die Elemente der Ma-

trix mit Werten besetzt werden können, die zu den in der Prämoralisierung vorkommenden Kanten korrespondieren, und diese Menge ist immer in der Menge der Kanten des Graphen enthalten. Dagegen darf beim PddP-Algorithmus die Diagonale mit negativen und positiven Elementen besetzt werden, beim Roverato-Ansatz dürfen es nur positive Elemente sein. Inwieweit sich diese Unterschiede auf das Optimierungsverfahren auswirken ist offen und kann den Startpunkt weiterer Untersuchungen bilden.

Zusammengefasst eignen sich sowohl der Ansatz von Roverato als auch der Ansatz mit Hilfe von prämoralisierbaren Graphen, um positiv definite Matrizen mit Nebenbedingungen zu erzeugen. Aber auch diagonaldominante Matrizen sind auf Grund der einfachen Handhabung eine Möglichkeit, sollten aber bei sehr großen Parametermengen in Simulationsansätzen keine Verwendung finden, da sie bei Knoten mit vielen Nachbarn keine großen partiellen Korrelationen darstellen können und zudem keine Zerlegung der finalen Matrix liefern, wie dies bei dem PddP-Ansatz und dem Ansatz von Roverato der Fall ist. Der Ansatz von Roverato ist für jeden Graphen einsetzbar, nicht nur für die prämoralisierbaren Graphen. Dieser Nachteil des PddP-Algorithmus tritt aber nicht gravierend bei Microarray-Daten in Erscheinung, da die Studien gezeigt haben, dass sehr viele der dort verwendeten Graphen prämoralisierbar sind. Allerdings nimmt der Anteil bei Graphen mit vielen Knoten signifikant ab (siehe Kapitel 5.2). Für Graphen mit sehr vielen Knoten ist somit ein zweigeteiltes Vorgehen sinnvoll. Für die Menge der nicht prämoralisierbaren Graphen wird der Ansatz von Roverato benutzt, während für die Menge der prämoralisierbaren Graphen auf Grund des Geschwindigkeitsvorteils beim Optimieren der PddP-Algorithmus eingesetzt wird.

Die Optimierung der Einträge der oberen Dreiecksmatrix beim PddP-Algorithmus ist in dieser Arbeit in Kapitel 4.4 behandelt worden. Es wurden für die Optimierung zwei sehr einfache Verfahren angewandt, das Nelder-Mead-Verfahren und das BFGS-Verfahren. Die Ergebnisse zeigen, dass man mit dem BFGS-Verfahren bessere Resultate erzielt als mit dem Nelder-Mead-Verfahren. Zudem weisen die durch das BFGS-Verfahren berechneten Matrizen einen geringen Abstand zu den vorgegebenen Matrizen auf, sowohl bei Vorgabe einer positiv definiten Matrix mit Nebenbedingungen, was eine Art notwendige Bedingung an die Methode darstellt, aber auch bei Vorgabe beliebiger Matrizen. Auf Grund dieser guten Ergebnisse sind in der Arbeit keine weiteren Optimierungsalgorithmen getestet worden, man kann aber natürlich auch jeden beliebigen anderen Optimierungsalgorithmus auf die in Kapitel 4 definierte Funktion OP_A anwenden, denn an die zu optimierenden Werte x und y müssen nur sehr allgemeine Voraussetzungen gestellt werden und trotzdem gelten für die resultierende Matrix automatisch die drei in Kapitel 4.4 geforderten Eigenschaften. Die mathematische Programmierung, die in Kapitel 4 kurz beschrieben worden ist, ist ein weiterer sehr erfolgsversprechender Ansatz bei Optimierungsproblemen und kann auch bei der Erzeugung positiv definiter Matrizen mit Nebenbedingungen genutzt werden. Ansätze, die die obere Dreiecksmatrix optimieren, haben gegenüber Optimierungsansätzen für die gesamte Matrix Ω aber immer den Vorteil, dass eine Zerlegung der Matrix mitgeliefert wird. Kombinationen zwischen solchen Ansätzen, wie dem PddP-Ansatz und aktuellen Optimierungsansätzen wie der mathematischen Programmierung, die für die Besetzung der

Elemente der oberen Dreiecksmatrix benutzt werden können, sind ein vielversprechender Ansatz zur Erstellung von positiv definiten Matrizen mit Nebenbedingungen.

Das Problem, eine positiv definite Matrix mit Nebenbedingungen zu finden, tritt nicht nur bei Simulationsansätzen auf. Neben der in Kapitel 5.1 untersuchten Fragestellung, ob es sinnvoller ist, ein Netzwerk auf Genebene oder auf Spotebene zu schätzen, wurde in Kapitel 5.4 dieser Arbeit ein Graphenvergleich auf Pathways durchgeführt. Ein neuer Ansatz mit Hilfe eines parametrischen Bootstraps wurde eingeführt und mit einem der wenigen in der Literatur vorhandenen Ansätze verglichen, welcher allerdings bei den untersuchten Pathways keine nutzbaren Ergbnisse erbrachte, da keiner der untersuchten Pathways signifikant war (siehe Kapitel 5.4). Für einen Ansatz mit Hilfe eines parametrischen Bootstraps muss die Kovarianzmatrix beziehungsweise die Präzessionsmatrix der gemeinsamen Verteilung geschätzt werden; in dieser Arbeit wurde hierfür wieder der Ansatz von Schäfer und Strimmer gewählt. Die Kovarianzmatrix Ω der gemeinsamen Verteilung wurde geschätzt und danach auf die signifikant von Null verschiedenen Elemente reduziert. Die so erzeugte Matrix $\bar{\Omega}$ war der Startpunkt der Simulationen. Wie die Untersuchungen in Kapitel 5.2 gezeigt haben, ist durch den beschriebenen Ansatz nicht sicher gestellt, dass die geschätzte Matrix $\bar{\Omega}$ noch positiv definit ist. Der Grund ist, dass, wenn eine positiv definite Matrix gegeben ist und man aus dieser Elemente auf Null setzt, die resultierende Matrix nicht notwendigerweise positiv definit ist (siehe hierzu auch die Einleitung von Kapitel 4). Es ist also möglich, dass $\bar{\Omega}$ nicht positiv definit ist, also muss man zu der Schätzung $\bar{\Omega}$ eine Matrix finden, die positiv definit ist und die Struktur von $\bar{\Omega}$ beibehält, um den parametrischen Bootstrap anwenden zu können. Dies ist wieder der Ansatzpunkt für den PddP-Algorithmus. Überraschenderweise ist dieser Fall aber in den zwei untersuchten Datensätzen nicht aufgetreten.

Das in Kapitel 5.4 eingeführte Verfahren lieferte im Experiment von Hannenhalli einige signifikante Pathways, die schon mit Kardiomyopathie in Verbindung gebracht worden sind; dies ist ein erfolgsversprechendes Ergebnis. Der parametrische Bootstrap sollte allerdings als ein erster Schritt gesehen werden, denn bei diesem Ansatz sind auch noch Punkte offen, die einer genaueren Untersuchung bedürfen. So gibt es in den beiden untersuchten Fragestellungen einen eindeutigen Trend, dass Pathways mit vielen Genen eine höhere Chance haben, signifikant zu sein. Eine Erklärung für dieses Verhalten ist, dass Graphen mit vielen Knoten viele zu betrachtende Variablen haben, und somit ist beim Testen die Power bei diesen Graphen größer als bei Graphen mit nur wenigen Knoten. Dieser Zusammenhang zwischen Knotenzahl und Power sollte noch genauer untersucht werden, auch mit Hilfe von Simulationsstudien. Auch die Ergebnisse bei der Betrachtung der Klassifikations-Gene sind nicht zufrieden stellend. Hier sollte untersucht werden, wie sich das Ergebnis gestaltet, wenn ein Klassifikator mit wenigen Genen ausgewählt wird. Zudem sollten andere Methoden als der Algorithmus von Schäfer und Strimmer zur Erzeugung der Netzwerke benutzt und mit den Ergebnissen dieser Arbeit verglichen werden, doch dies gestaltet sich im Moment schwierig, da andere Algorithmen viel mehr Rechenzeit benötigen. Bei einem in dieser Arbeit nicht näher vorgestellten Ansatz von Castelo[12] benötigt man für einen einzelnen Graphen schon 10 Minuten, und ähnliche Zeiten sind beim Ansatz von Dobra zu

erwarten [64]. Im Moment stellt der Schäfer-Strimmer-Algorithmus somit eine der wenigen Möglichkeiten dar, den parametrischen Bootstrap anzuwenden, was sich in der Zukunft durch schnellere Implementierungen und Computer sicher ändern wird.

Für die Schätzung der gemeinsamen Verteilung können die Ansätze von Castelo, Dobra oder Meinshausen aber schon angewandt werden, denn diese muss nur einmal durchgeführt werden und aus diesem Grund ist der Zeitunterschied nicht so gravierend. Der Einsatz diser Methoden wäre eine sinnvolle Modifikation des Vorgehens, um zu überprüfen, ob die Ergebnisse robust sind. Eine andere Möglichkeit besteht darin, die Schranke für das Hinzufügen einer Kante zu ändern. Um dies zu testen, ist der Ansatz des parameterischen Bootstraps beim Datensatz von Hannenhalli auch mit der q-Wert-Schranke 0.05 getestet worden. Die Ergebnisse waren sehr ähnlich zu denen mit einer q-Wert-Schranke von 0.2. Mit einem Schrankenwert von 0.2 gibt es 20 Pathways mit einem adjustierten p-Wert von 0.05, bei einer Schranke von 0.05 sind es 18 Pathways. 17 Pathways sind bei beiden Untersuchungen signifikant. Dies gibt einen Hinweis darüber, dass die Untersuchungen robust gegenüber der Schrankenwahl sind.

Bei dem parametrischen Bootstrap wird für die Nullhypothese angenommen, dass die Daten aus einer gemeinsamen Normalverteilung generiert werden. Möchte man nicht solche strengen Voraussetzungen machen, oder sind diese nicht erfüllt, so können auch andere Ansätze zum Strukturvergleich benutzt werden. Beispielsweise ist auch ein Permutationsansatz denkbar, bei dem die Daten in zufällige Untergruppen aufgeteilt werden und aus diesen ein Netzwerk geschätzt wird. Bei einem solchen Ansatz geht man nur von der Austauschbarkeit der Datenpunkte aus. Allerdings ist es mit dem parametrischen Bootstrap möglich, feinere Strukturunterschiede zu detektieren. Zwischen diesen Ansätzen gibt es also Unterschiede, die weitere Untersuchungen nach sich ziehen können. Zusammengenommen verbleiben noch viele Fragen, was die Vergleichbarkeit von Strukturen bei Microarray-Daten betrifft, und somit ist Kapitel 5.4 dieser Arbeit als eine Art Ausblick und Anregung gedacht, diesen Fragen nachzugehen, bei deren Beantwortung das Erzeugen von positiv definiten Matrizen mit Nebenbedingungen und vielleicht auch prämoralisierbare Graphen eine Rolle spielen werden, falls man Gaußsche graphische Modelle nutzen möchte.

Anhang A

Das R-Paket graphiti

Alle Berechnungen in dieser Arbeit wurden in R[37] und Bioconductor[27] durchgeführt. In R ist es mit Hilfe von Paketen und Vignette möglich, Dokumente zu erstellen, die die durchgeführten Analysen reproduzierbar machen [62, 42]. So basiert auch diese Arbeit auf einer Ansammlung von Vignetten, die vor allem das R-Paket *graphiti* nutzen.
In *graphiti* sind die Algorithmen implementiert, die in dieser Arbeit beschrieben worden sind. Den Kernpunkt bilden die Funktionen zur Erstellung einer Prämoralisierung für ungerichtete Graphen. Der Algorithmus I aus Kapitel 3 ist als Funktion `decPraeMor` implementiert und Algorithmus II als Funktion `praeMor`. Die Funktion `PraeMoralise` verbindet die zwei Algorithmen. Es wird hier separat für jede Zusammenhangskomponente eines Graphen entweder Algorithmus I oder II angewandt, je nachdem ob die Komponente zerlegbar ist oder nicht. Dieses Vorgehen ist in Kapitel 3 motiviert. Die Eingabe für jede der drei Funktionen ist ein ungerichteter Graph H. Dieser Graph ist ein Element der *graph-NEL* Klassse, die im R-Paket *graph*[28] definiert wird. Die Anwendung der Funktionen soll hier am Beispiel von `praeMor` dargestellt werden. Man startet mit einer Matrix, die als Adjazenzmatrix fungiert und erstellt so einen ungerichteten Graphen H. In diesem Beispiel werden zwei Graphen erzeugt.

```
> library(graph)
> library(graphiti)
> W1 <- matrix(c(0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0,
+     0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1,
+     0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0), ncol = 7)
> rownames(W1) <- colnames(W1) <- letters[1:7]
> W2 <- matrix(c(0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0),
+     ncol = 4)
> rownames(W2) <- colnames(W2) <- letters[1:4]
> GraphH1 <- as(W1, "graphNEL")
> GraphH2 <- as(W2, "graphNEL")
```

```
> GraphH1

A graphNEL graph with undirected edges
Number of Nodes = 7
Number of Edges = 10
```

Der erste Graph entspricht dem Graphen aus Abbildung 3.3. Dieser Graph ist nicht zerlegbar. Man muss also die Funktion `praeMor` anwenden, um eine Prämoralisierung zu bekommen, falls eine existiert.

```
> GraphG <- praeMor(GraphH1)

> GraphG

A graphNEL graph with directed edges
Number of Nodes = 7
Number of Edges = 8
```

Der zweite Graph ist der 4-Zykel ohne Abkürzung (siehe Abbildung 3.1, Bild A).

```
> GraphH2

A graphNEL graph with undirected edges
Number of Nodes = 4
Number of Edges = 4
```

Wie gezeigt gibt es für Zykel ohne Abkürzung keine Prämoralisierung, der Algorithmus gibt in diesem Fall eine Warnung und einen Graphen ohne Kanten zurück.

```
> Versuch <- praeMor(GraphH2)

Graph cannot be premoralised

> Versuch

A graphNEL graph with directed edges
Number of Nodes = 4
Number of Edges = 0
```

Es besteht die Möglichkeit zu überprüfen, ob ein gegebener gerichteter Graph eine Prämoralisierung eines ungerichteten Graphen ist. Dies geschieht mit Hilfe der Funktion `Moral`.

```
> Moral(Versuch, GraphH2)

[1] FALSE
```

```
> Moral(GraphG, GraphH1)

[1] TRUE
```

In Kapitel 4 wird der PddP-Algorithmus vorgestellt, mit dessen Hilfe man eine positiv definite Matrix erzeugen kann, die bezüglich der Frobeniusnorm einen möglichst kleinen Abstand zu einer vorgegebenen Matrix besitzt. Die Funktion `GenerateMatrices` beruht auf diesem Algorithmus. In dieser Funktion gibt man sich eine Matrix A vor, die die partiellen Korrelationen beschreiben soll. Die Funktion generiert dann eine positiv definite Matrix, die als normierte Präzessionsmatrix fungiert (sihe Kapitel 4). In diesem Beispiel wird die Matrix $W1$ benutzt, wobei an Stelle der 1en zufällige Werte erzeugt werden.

```
> Kanten <- sum(W1 == 1)/2
> W <- sample(c(-1, 1), Kanten, replace = TRUE) * runif(Kanten,
+     min = 0.5, max = 1)
> W1[lower.tri(W1)] <- 0
> W1[W1 == 1] <- W
> W1[lower.tri(W1)] <- t(W1)[lower.tri(W1)]
> diag(W1) <- 1
> Result <- GenerateMatrices(W1, metrik1, method = "BFGS")
```

`Result` gibt die geschätzte Matrix Ω zurück, wobei hier der Vorzeichenwechsel zu beachten ist (siehe Kapitel 4).

```
> round(W1, 2)

      a     b     c     d     e     f     g
a  1.00 -0.52 -0.57 -0.60  0.00  0.00  0.00
b -0.52  1.00  0.00  0.00  0.75 -0.76  0.00
c -0.57  0.00  1.00  0.00  0.55  0.00  0.67
d -0.60  0.00  0.00  1.00 -0.65  0.00  0.00
e  0.00  0.75  0.55 -0.65  1.00  0.97 -0.52
f  0.00 -0.76  0.00  0.00  0.97  1.00  0.00
g  0.00  0.00  0.67  0.00 -0.52  0.00  1.00

> round(Result$matrix, 2)

     a     b     c    d     e     f     g
a 1.00  0.33  0.41 0.42  0.00  0.00  0.00
b 0.33  1.00  0.00 0.00 -0.58  0.77  0.00
c 0.41  0.00  1.00 0.00 -0.32  0.00 -0.69
d 0.42  0.00  0.00 1.00  0.36  0.00  0.00
e 0.00 -0.58 -0.32 0.36  1.00 -0.75  0.46
f 0.00  0.77  0.00 0.00 -0.75  1.00  0.00
g 0.00  0.00 -0.69 0.00  0.46  0.00  1.00
```

Aber `Result` beinhaltet nicht nur die erzeugte Matrix, sondern auch die obere Dreiecksmatrix, mit deren Hilfe man, wie im Algorithmus A in Kapitel 4 beschrieben, Daten aus einer multivariaten Normalverteilung erzeugen kann. Dieser Algorithmus, inklusive einer Festlegung der partiellen Varianzen, wird im folgenden Programmcode genutzt.

```
> Q <- Result$matrix
> AnzahlGene <- nrow(Q)
> D <- diag(sqrt(abs(rnorm(AnzahlGene, 1, 5))))
> Dreiecksmatrix <- Result$upperTri
> pVarMatrixsortiert <- D[order(Result$order), order(Result$order)]
> erzDaten <- c()
> for (i in 1:10000) {
+     z <- rnorm(AnzahlGene)
+     x <- backsolve(Dreiecksmatrix %*% pVarMatrixsortiert, z)
+     erzDaten <- cbind(erzDaten, x)
+ }
> erzDaten <- erzDaten[Result$order, ]
> rownames(erzDaten) <- paste("G", 1:nrow(erzDaten), sep = "_")
> colnames(erzDaten) <- paste("Sample", 1:ncol(erzDaten), sep = "_")
```

Das Objekt *erzDaten* ist eine Matrix mit 7 Zeilen und 10000 Spalten. Jede Spalte ist die Realisierung einer multivariaten Normalverteilung zum Mittelwert 0 und Kovarianzmatrix $\Sigma = (D \cdot Q \cdot D)^{-1}$. Hierbei ist Q die negative partielle Korrelationsmatrix. Dies erkennt man auch, wenn man die empirische Kovarianzmatrix aus den erzeugten Daten schätzt und diese in die negative partielle Korrelationsmatrix umwandelt.

```
> round(cov2cor(solve(cov(t(erzDaten)))), 2)
```

```
     G_1   G_2   G_3  G_4   G_5   G_6   G_7
G_1 1.00  0.34  0.40 0.42  0.01  0.00  0.01
G_2 0.34  1.00  0.00 0.01 -0.57  0.77  0.01
G_3 0.40  0.00  1.00 0.00 -0.32  0.00 -0.69
G_4 0.42  0.01  0.00 1.00  0.36  0.00  0.00
G_5 0.01 -0.57 -0.32 0.36  1.00 -0.75  0.46
G_6 0.00  0.77  0.00 0.00 -0.75  1.00  0.00
G_7 0.01  0.01 -0.69 0.00  0.46  0.00  1.00
```

```
> round(Q, 2)

     a     b     c    d     e     f     g
a 1.00  0.33  0.41 0.42  0.00  0.00  0.00
b 0.33  1.00  0.00 0.00 -0.58  0.77  0.00
c 0.41  0.00  1.00 0.00 -0.32  0.00 -0.69
d 0.42  0.00  0.00 1.00  0.36  0.00  0.00
e 0.00 -0.58 -0.32 0.36  1.00 -0.75  0.46
f 0.00  0.77  0.00 0.00 -0.75  1.00  0.00
g 0.00  0.00 -0.69 0.00  0.46  0.00  1.00
```

Anhang B

Bereitstellung der Datensätze

Jeder in dieser Arbeit benutzte Datensatz lag für die Untersuchung als ein `expression set` vor, das ist ein Datentyp in R. Dieser Datentyp enthält nicht nur die Expressionswerte der einzelnen Chips (`exprs`), sondern auch Informationen über die Proben/Patienten (`phenoData`). In Tabelle B.1 ist für jeden Datensatz der Name der Spalte des zugehörigen `phenoData` Objektes angegeben, die für den Vergleich benutzt worden ist.

	Vergleichsspalte	Gruppe 1	Gruppe 2
Wang	ER.Status	ER-	ER+
Hannenhalli	diagnosis	Idiopathic	Ischemic
Bhattacharjee	Class	ADENO	SQUAMOUS
Gruvberger	ER status	-	+
Nevins	LN.status	neg	pos
Nevins	ER.status	neg	pos
Beer	Type	Normal	Tumor
Garber	AC.SCC	AC	SCC
Barth/Kuner A	Cy3.Diagnosis	DilatedCardiomyopathy	IschemicCardiomyopathy
Barth/Kuner B	Cy3.Diagnosis	DilatedCardiomyopathy	Normal

Tabelle B.1: Überblick der Datensätze

Die notwendige Annotation der einzelnen Datensätze wurde in mehreren Schritten durchgeführt. Zuerst wurde das R-Paket *KEGG*[46] genutzt, um für jeden KEGG Pathway die Entrez IDs der Gene zu ermitteln, die Teil des Pathways sind. Dann wurde für jeden Datensatz den eindeutigen Spot-Identifiern (*affy ID,Acession number, CloneID*) eine Entrez Gene ID zugeordnet:

- **Wang** Affymetrix Datensatz. Das Bioconductor-Annotationspaket *hgu133a*[43] (Version 1.14.0) wurde für die Zuordnung benutzt.
- **Nevins** Affymetrix Datensatz. Das Bioconductor-Annotationspaket *hu6800*[45] (Version 1.14.0) wurde für die Zuordnung benutzt.

- **Bhattacharjee** Affymetrix Datensatz. Das Bioconductor-Annotationspaket *hgu95av2* [44] (Version 1.14.0) wurde für die Zuordnung benutzt.
- **Gruvberger** cDNA Datensatz. Die Zuordnung erfolgte mit Hilfe des Webtools SOURCE (http://source.stanford.edu/cgi-bin/source/sourceSearch Stand 15.01.2007). Die Identifier des Datensatzes sind *CloneIDs*.
- **Beer** Affymetrix Datensatz. Das Bioconductor-Annotationspaket *hu6800*[45] (Version 1.14.0) wurde für die Zuordnung benutzt.
- **Hannenhalli** Affymetrix Datensatz. Das Bioconductor-Annotationspaket *hgu133a* [43] (Version 1.14.0) wurde für die Zuordnung benutzt.
- **Garber** cDNA Datensatz. Die Zuordnung erfolgte mit Hilfe des Webtools SOURCE (http://source.stanford.edu/cgi-bin/source/sourceSearch Stand 15.01.2007). Die Identifier des Datensatzes sind *Accession numbers*.
- **Barth/Kuner A + B** cDNA Datensatz. Zuordnung erfolgte mit Hilfe einer DKFZ-internen Datenbank (Stand 15.01.2007). Die Identifier des Datensatzes sind *RZPD IDs*.

Somit hat man für jeden Pathway und jeden Datensatz eine Liste der Gene erhalten, die sich im Pathway, aber auch auf dem für den Datensatz benutzten Microarray befinden. Die folgenden Tabellen enthalten für jeden Pathway mit mehr als 30 Genen die Anzahl der Gene, die sich auf dem Affymetrix-Chip *hgu133a* befinden. Dieser Chiptyp liegt allen in Kapitel 5.4 benutzten Datensätzen zu Grunde. Zusätzlich ist in dieser Tabelle für jeden Pathway die Abkürzung aufgeführt, die in den Tabellen 5.8,5.9,5.10,5.11 und 5.12 benutzt worden ist.

	Name	Genzahl	Abkürzung
1	Linoleic acid metabolism	31	Linoleic ac..ism
2	Sphingolipid metabolism	31	Sphingolipi..ism
3	Proteasome	31	Proteasome
4	SNARE interactions in vesicular transport	32	SNARE inter..ort
5	Basal transcription factors	32	Basal trans..ors
6	Folate biosynthesis	33	Folate bios..sis
7	Propanoate metabolism	33	Propanoate ..ism
8	Neurodegenerative Disorders	34	Neurodegene..ers
9	Nicotinate and nicotinamide metabolism	34	Nicotinate ..ism
10	ATP synthesis	34	ATP synthes
11	Bile acid biosynthesis	34	Bile acid b..sis
12	Histidine metabolism	35	Histidine m..ism
13	Glutathione metabolism	35	Glutathione..ism
14	N-Glycan biosynthesis	35	N-Glycan bi..sis
15	Taste transduction	35	Taste trans..ion
16	ABC transporters - General	36	ABC transpo..ral
17	Pyruvate metabolism	37	Pyruvate me..ism
18	Cholera - Infection	37	Cholera - I..ion
19	Fructose and mannose metabolism	37	Fructose an..ism
20	Notch signaling pathway	39	Notch signa..way
21	Glycine, serine and threonine metabolism	39	Glycine, se..ism
22	Butanoate metabolism	40	Butanoate m..ism
23	Ubiquitin mediated proteolysis	40	Ubiquitin m..sis
24	Type I diabetes mellitus	42	Type I diab..tus
25	Type II diabetes mellitus	42	Type II dia..tus
26	Androgen and estrogen metabolism	42	Androgen an..ism
27	Lysine degradation	45	Lysine degr..ion
28	Hedgehog signaling pathway	45	Hedgehog si..way
29	Fatty acid metabolism	46	Fatty acid ..ism
30	Inositol phosphate metabolism	47	Inositol ph..ism

Tabelle B.2: Annotation Pathways

	Name	Genzahl	Abkürzung
1	mTOR signaling pathway	47	mTOR signal..way
2	Valine, leucine and isoleucine degradation	49	Valine, leu..ion
3	Pathogenic Escherichia coli infection - EHEC	49	Pathogenic ..HEC
4	Pathogenic Escherichia coli infection - EPEC	49	Pathogenic ..PEC
5	Arginine and proline metabolism	50	Arginine an..ism
6	Arachidonic acid metabolism	50	Arachidonic..ism
7	Glycerolipid metabolism	51	Glycerolipi..ism
8	Tyrosine metabolism	52	Tyrosine me..ism
9	Glycan structures - biosynthesis 2	54	Glycan stru..s 2
10	Glycolysis / Gluconeogenesis	56	Glycolysis ..sis
11	Metabolism of xenobiotics by cytochrome P450	56	Metabolism ..450
12	B cell receptor signaling pathway	60	B cell rece..way
13	Starch and sucrose metabolism	61	Starch and ..ism
14	PPAR signaling pathway	63	PPAR signal..way
15	Epithelial cell signaling in Helicobacter pylori infection	64	Epithelial ..ion
16	Long-term potentiation	65	Long-term p..ion
17	Glycerophospholipid metabolism	66	Glycerophos..ism
18	Adipocytokine signaling pathway	66	Adipocytoki..way
19	Complement and coagulation cascades	68	Complement ..des
20	VEGF signaling pathway	68	VEGF signal..way
21	Tryptophan metabolism	70	Tryptophan ..ism
22	Colorectal cancer	72	Colorectal ..cer
23	Long-term depression	73	Long-term d..ion
24	Pyrimidine metabolism	74	Pyrimidine ..ism
25	Adherens junction	74	Adherens ju..ion
26	Fc epsilon RI signaling pathway	74	Fc epsilon ..way
27	Phosphatidylinositol signaling system	74	Phosphatidy..tem
28	TGF-beta signaling pathway	79	TGF-beta si..way
29	Apoptosis	81	Apoptosis
30	Antigen processing and presentation	82	Antigen pro..ion

Tabelle B.3: Annotation Pathways (Fortsetzung 1)

	Name	Genzahl	Abkürzung
1	ECM-receptor interaction	83	ECM-recepto..ion
2	Glycan structures - biosynthesis 1	83	Glycan stru..s 1
3	Hematopoietic cell lineage	87	Hematopoiet..age
4	Toll-like receptor signaling pathway	89	Toll-like r..way
5	T cell receptor signaling pathway	90	T cell rece..way
6	GnRH signaling pathway	93	GnRH signal..way
7	Ribosome	93	Ribosome
8	Gap junction	95	Gap junctio
9	Oxidative phosphorylation	103	Oxidative p..ion
10	Cell cycle	103	Cell cycle
11	Cell Communication	106	Cell Commun..ion
12	Tight junction	107	Tight junct
13	Leukocyte transendothelial migration	108	Leukocyte t..ion
14	Axon guidance	116	Axon guidan
15	Natural killer cell mediated cytotoxicity	119	Natural kil..ity
16	Cell adhesion molecules (CAMs)	120	Cell adhesi..Ms)
17	Wnt signaling pathway	129	Wnt signali..way
18	Insulin signaling pathway	129	Insulin sig..way
19	Purine metabolism	130	Purine meta..ism
20	Jak-STAT signaling pathway	140	Jak-STAT si..way
21	Calcium signaling pathway	165	Calcium sig..way
22	Focal adhesion	186	Focal adhes
23	Regulation of actin cytoskeleton	187	Regulation ..ton
24	Cytokine-cytokine receptor interaction	235	Cytokine-cy..ion
25	MAPK signaling pathway	256	MAPK signal..way
26	Neuroactive ligand-receptor interaction	272	Neuroactive..ion

Tabelle B.4: Annotation Pathways (Fortsetzung 2)

Anhang C

Pathway Bilder

Im Folgenden sind einige im Herzdatensatz von Hannenhalli und im Brustkrebsdatensatz von Wang als signifikant gefundene Pathways (siehe Kapitel 5.4) abgebildet. In jeder Studie wurden zwei Untergruppen miteinander verglichen. Im der Hannenhalli-Studie besteht die Untergruppe 1 aus den Proben von Patienten mit dilatativer Kardiomyopathie(DCM) und Untergruppe 2 aus den Proben von Patienten mit ischämischer Kardiomyopathie(ICM). Im Brustkrebsdatensatz besteht Untergruppe 1 aus den Patienten mit negativem Estrogen-Rezeptor-Status und Untergruppe 2 aus den Patienten mit positivem Estrogen-Rezeptor-Status. Um die Unterschiede in der partiellen Korrelationsstruktur zwischen den jeweils zwei untersuchten Untergruppen besser erkennen zu können, werden beide Strukturen in einem Graphen dargestellt. Hierbei gibt der Typ der Kante Aussage darüber, ob diese nur in einer der beiden Untergruppen oder in beiden gefunden worden ist:

- **gepunktete Linie**: nur in Untergruppe 1 als signifikant gefunden
- **gestrichelte Linie**: nur in Untergruppe 2 als signifikant gefunden
- **durchgezogene Linie**: in beiden Gruppen als signifikant gefunden

Zusätzlich zeigt eine Farbcodierung der Kanten an, ob es sich um eine positive oder negative partielle Korrelation handelt, oder ob, falls die Kante in beiden Untergruppen gefunden wurde, in den zwei Gruppen gegenläufige Korrelationen gefunden wurden. Dies tritt aber nur sehr selten auf.

- **schwarz:** positive partielle Korrelation in der Untergruppe oder in beiden Untergruppen
- **rot:** negative partielle Korrelation in der Untergruppe oder in beiden Untergruppen
- **grün:** positive partielle Korrelation in Untergruppe 1 und negative partielle Korrelation in Untergruppe 2
- **blau:** positive partielle Korrelation in Untergruppe 2 und negative partielle Korrelation in Untergruppe 1

Auch die Knoten werden mit verschiedenen Farben versehen. Diese richten sich danach, ob das durch einen Knoten repräsentierte Gen in einem SAM[72] Vergleich zwischen den Untergruppen als signifikant unterschiedlich exprimiert gefunden worden ist und in welcher Untergruppe dieses Gen höher exprimiert ist. Blaue Knoten kennzeichnen hierbei die Gene, die in der Untergruppe 2 eine höhere Expression haben als in der Untergruppe 1. Rote Knoten kennzeichnen die Gene, die in der Untergruppe 1 eine höhere Expression haben als in der Untergruppe 2. Die genaue Farbwahl der Knoten bei verschiedenen definierten Schwellenwerten des q-Wertes aus SAM ist in Abbildung C.1 ersichtlich. Die Beschriftung der Knoten ist eine Kombination aus Gensymbol (falls vorhanden) und Entrez ID (gekennzeichnet durch das Präfix *EID*).

Abbildung C.1: Die Farbabstufung bei vorgegebenen Schwellenwerten des q-Wertes in SAM.

Der so in R erzeugte Graph kann mit Hilfe einer Funktion des *graphiti* Paketes als *.graphml* abgespeichert werden und dann beispielsweise mit dem frei erhältlichen java Programm *yed* (http://www.yworks.com/en/products_yed_about.htm) (basierend auf der java Bibliothek yFiles) geöffnet werden. In dem Programm *yed* kann ein Graph noch bearbeitet und in verschiedenen Formaten (beispielsweise .pdf) abgespeichert werden. Man erkennt an den folgenden Bildern auch sehr deutlich, dass sich zwischen den Untergruppen im Hannenhalli-Datensatz nicht viele differentielle Gene befinden, wohl aber zwischen den Untergruppen im Wang-Datensatz.

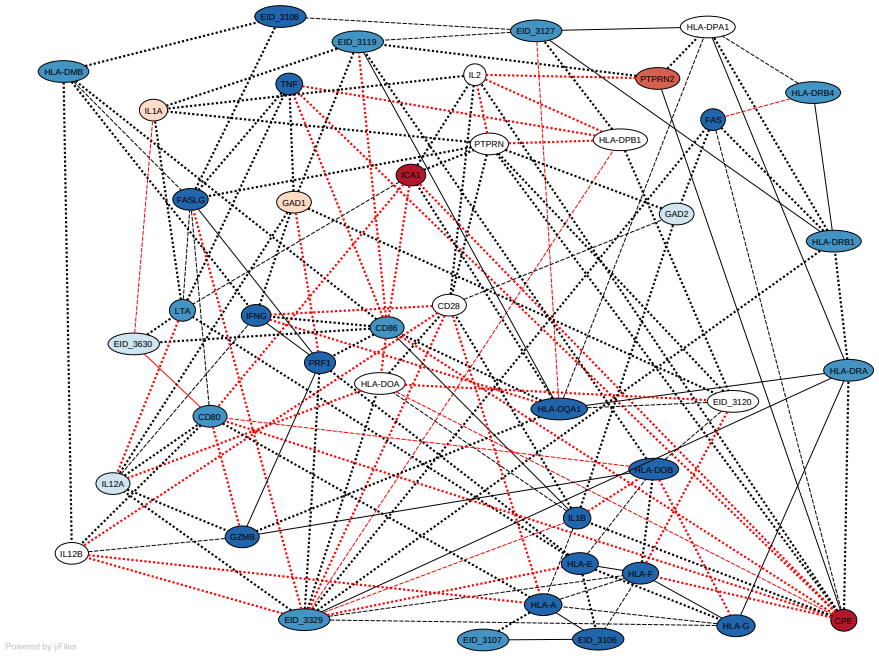

Abbildung C.2: Der *Type I diabetes mellitus* Pathway im Wang-Datensatz.

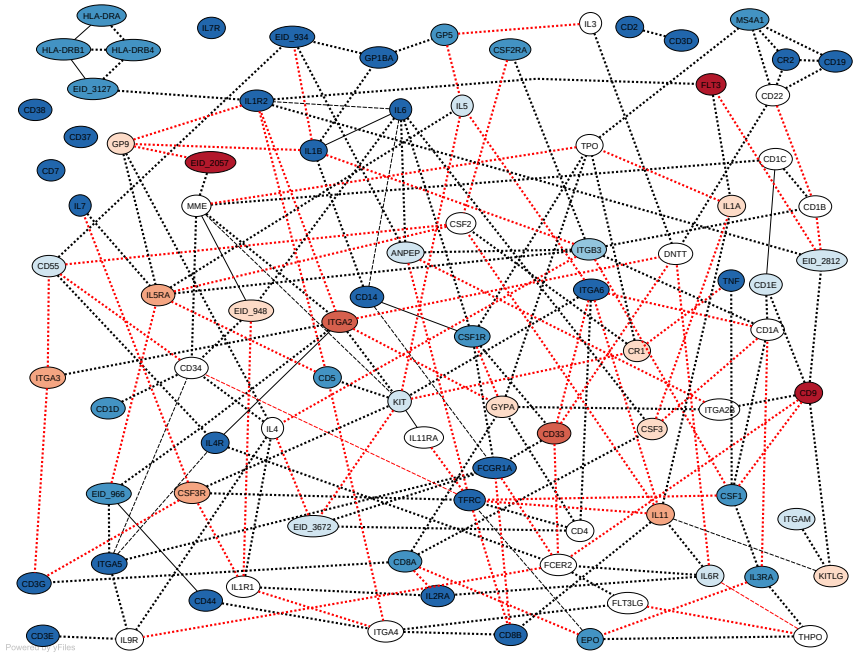

Abbildung C.3: Der *Hematopoietic cell lineage* Pathway im Wang-Datensatz.

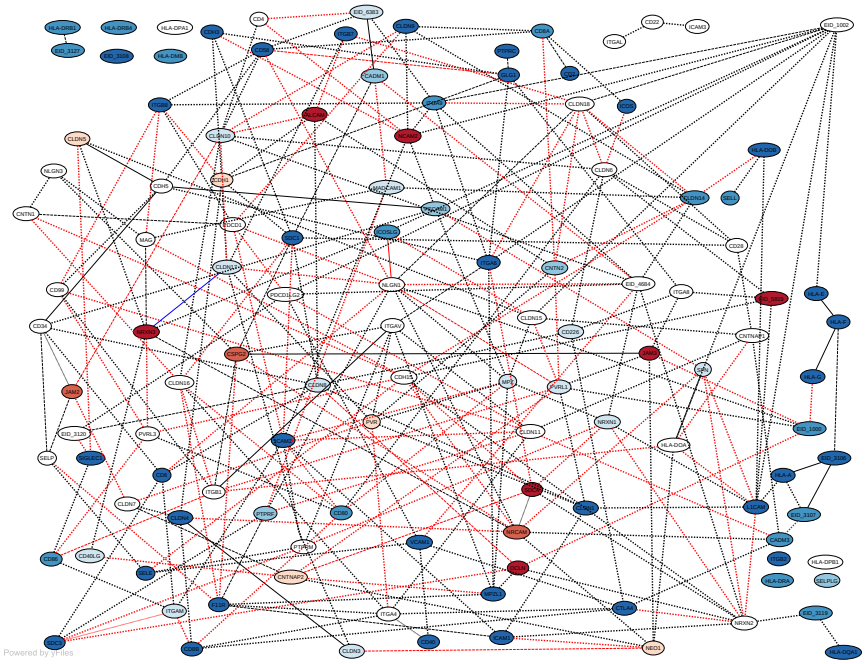

Abbildung C.4: Der *Cell adhesion* Pathway im Wang-Datensatz.

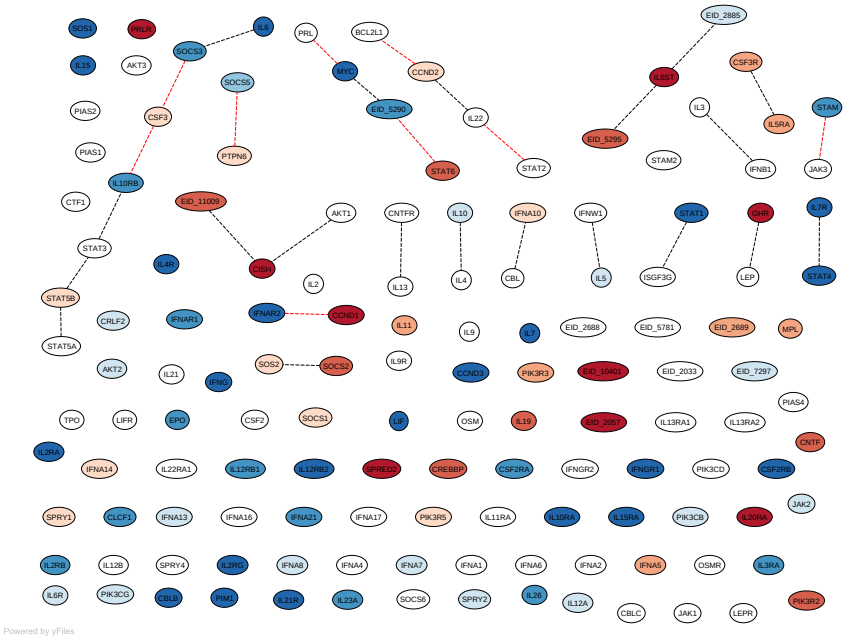

Abbildung C.5: Der *JAK-STAT signaling* Pathway im Wang-Datensatz.

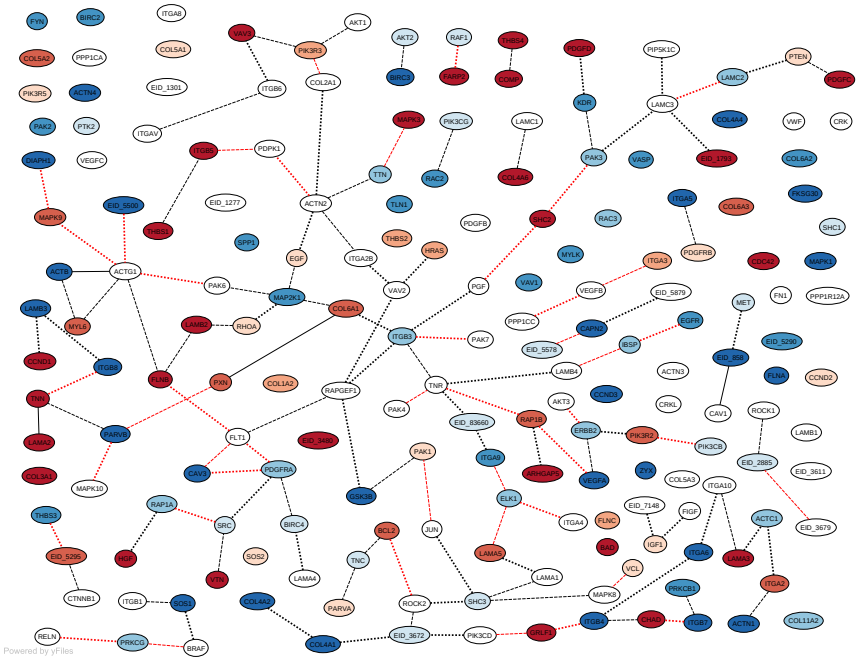

Abbildung C.6: Der *Focal adhesion* Pathway im Wang-Datensatz.

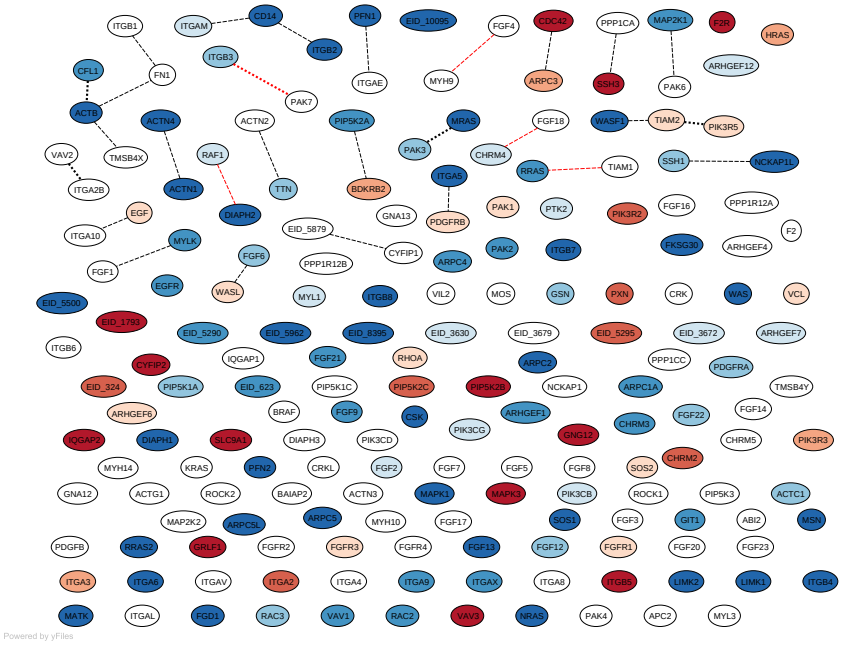

Abbildung C.7: Der *Regulation of actin cytoskeleton* Pathway im Wang-Datensatz.

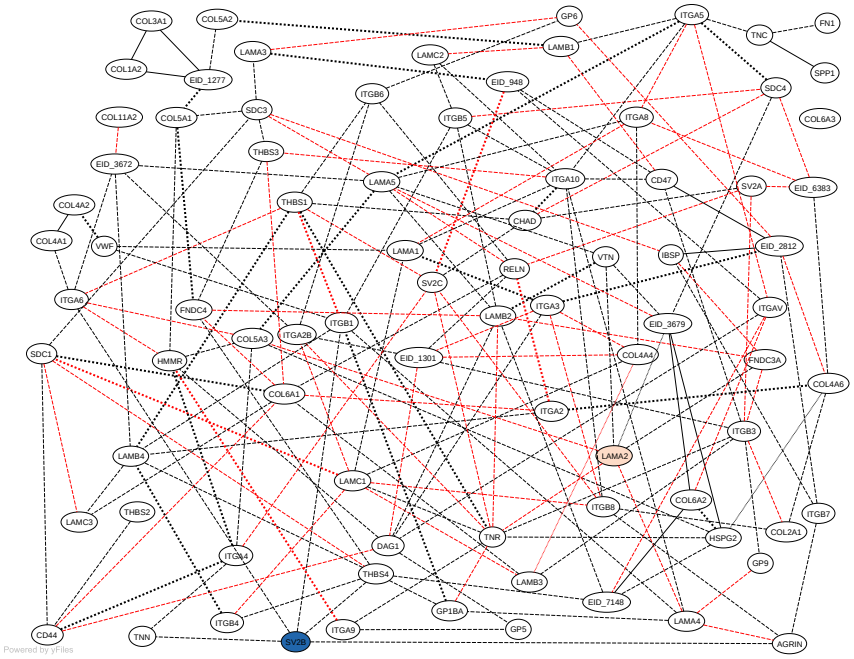

Abbildung C.8: Der *ECM receptor interaction* Pathway im Hannenhalli-Datensatz.

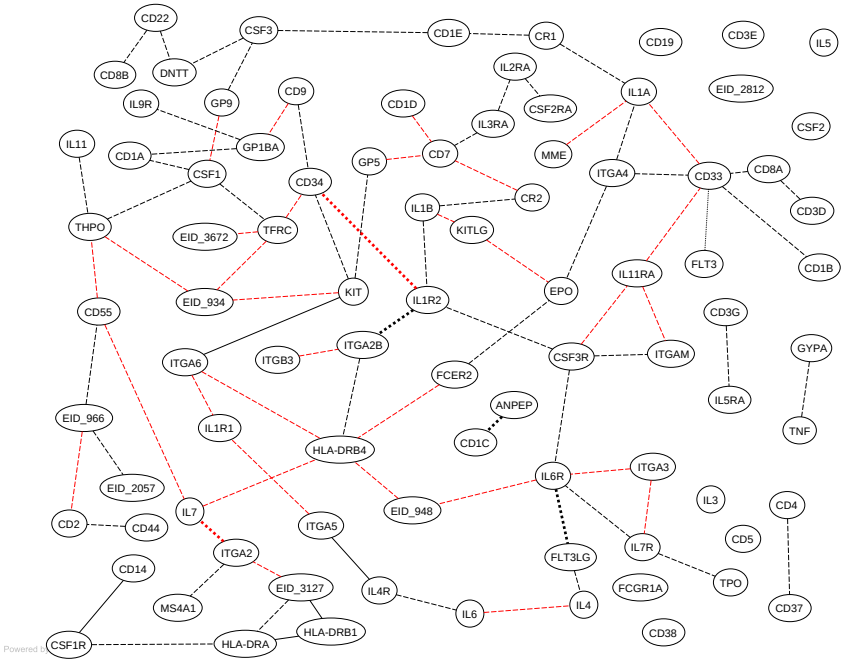

Abbildung C.9: Der *Hematopoietic cell lineage* Pathway im Hannenhalli-Datensatz.

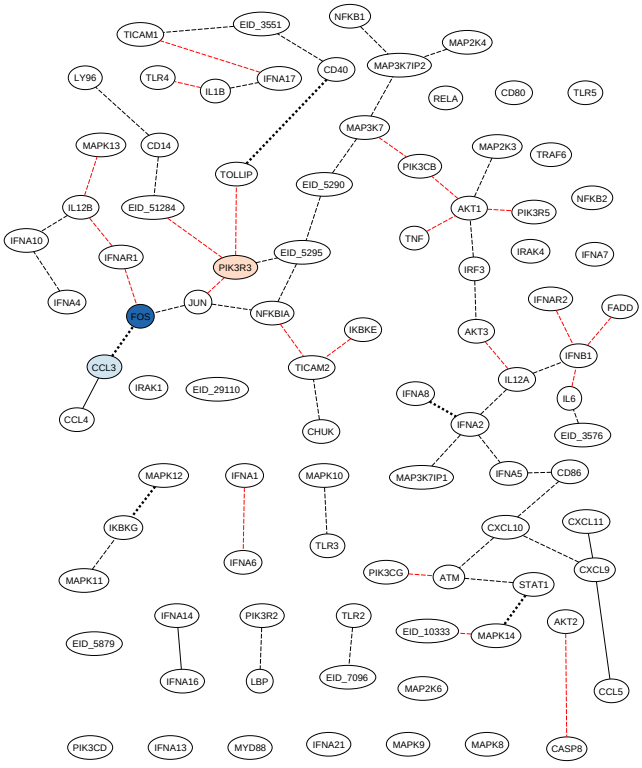

Abbildung C.10: Der *toll-like receptor signaling* Pathway im Hannenhalli-Datensatz.

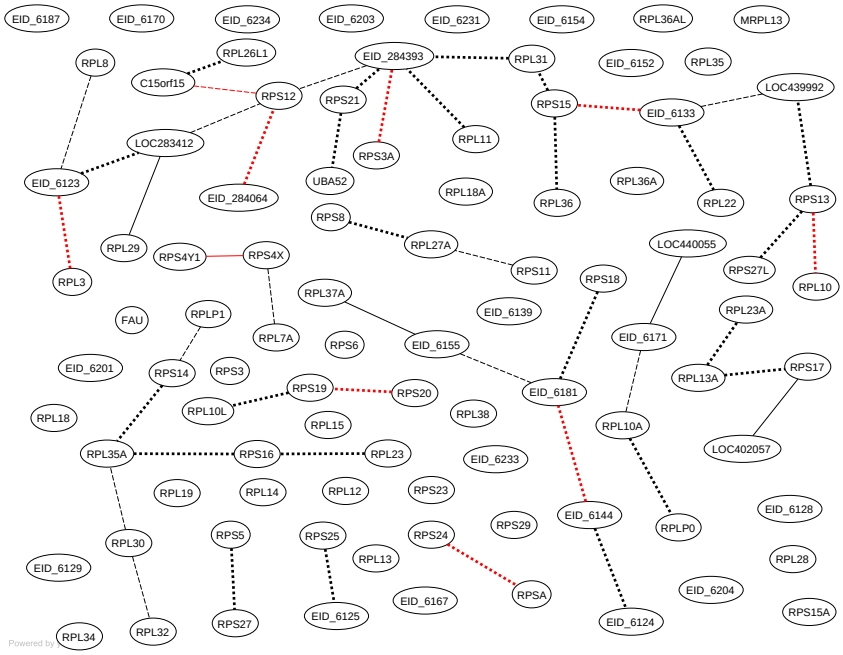

Abbildung C.11: Der *Ribosome* Pathway im Hannenhalli-Datensatz.

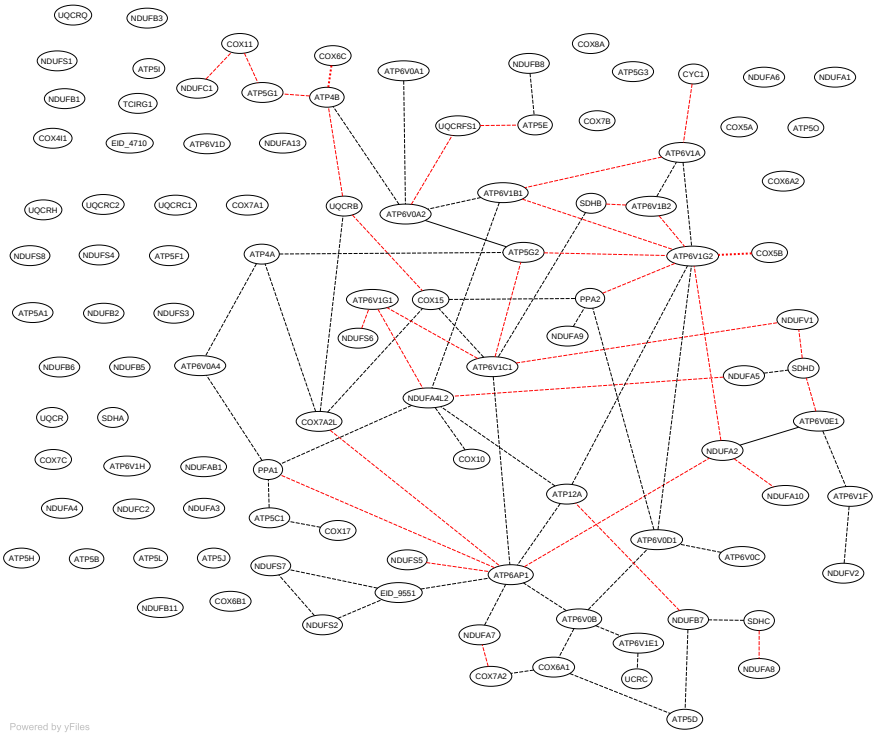

Abbildung C.12: Der *oxidative phosphorylation* Pathway im Hannenhalli-Datensatz.

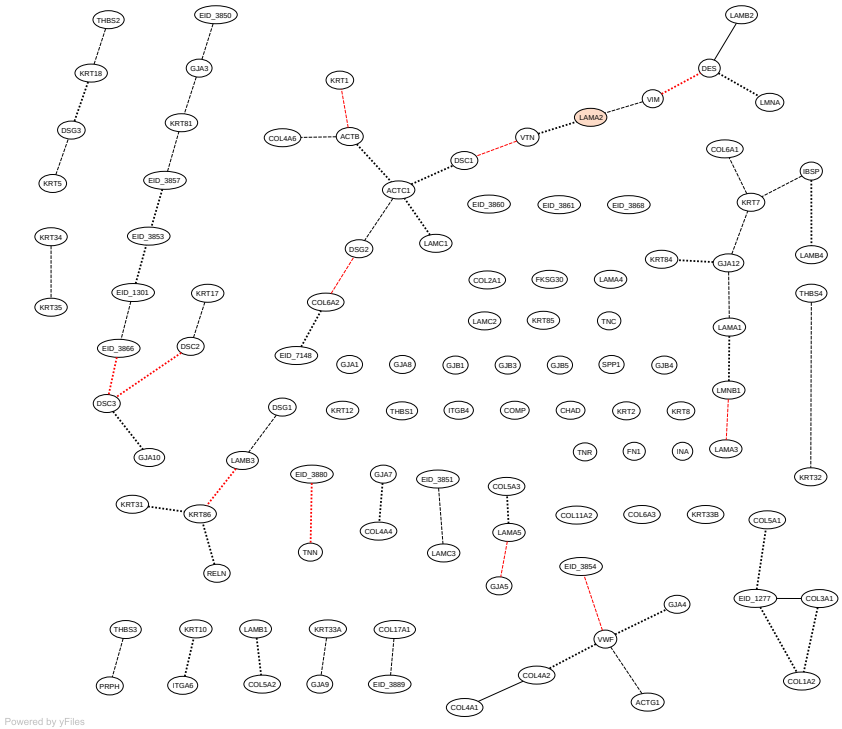

Abbildung C.13: Der *cell communication* Pathway im Hannenhalli-Datensatz.

Literaturverzeichnis

[1] F Alla, S Briancon, Y Juilliere, PM Mertes, JP Villemot, und F Zannad. Differential clinical prognostic classifications in dilated and ischemic advanced heart failure: the epical study. *Am Heart J*, 139:895–904, 2000.

[2] R Balasubramanian, T LaFramboise, D Scholtens, und R Gentleman. A graph-theoretic approach to testing associations between disparate sources of functional genomics data. *Bioinformatics*, 20(18):3353–3362, 2004.

[3] AS Barth, R Kuner, A Buness, M Ruschhaupt, S Merk, L Zwermann, S Kaab, E Kreuzer, G Steinbeck, U Mansmann, A Poustka, M Nabauer, und H Sültmann. Identification of a common gene expression signature in dilated cardiomyopathy across independent microarray studies. *J. Am. Coll. Cardiol.*, 48(8):1610–1617, 2006.

[4] DG Beer, SL Kardia, CC Huang, TJ Giordano, AM Levin, DE Misek, L Lin, G Chen, TG Gharib, DG Thomas, ML Lizyness, R Kuick, S Hayasaka, JM Taylor, MD Iannettoni, MB Orringer, und S Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nat Med*, 8(8):816–824, 2002.

[5] Y Benjamini und D Yekutieli. The control of the false discovery rate in multiple hypothesis testing under dependency. *The Annals of Statistics*, 29(4):1165–1188, 2001.

[6] KP Bennett und E Parrado-Hernandez. The interplay of optimization and machine learning research. *Journal of Machine Learning Research*, 7:1265–1281, 2006.

[7] A Bhattacharjee, WG Richards, J Staunton, C Li, S Monti, P Vasa, C Ladd, J Beheshti, R Bueno, M Gillette, M Loda, G Weber, EJ Mark, ES Lander, W Wong, BE Johnson, TR Golub, DJ Sugarbaker, und M Meyerson. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc. Natl. Acad. Sci. U.S.A.*, 98(24):13790–13795, 2001.

[8] JH Boyd, S Mathur, Y Wang, RM Bateman, und KR Walley. Toll-like receptor stimulation in cardiomyoctes decreases contractility and initiates an nf-kappab dependent inflammatory response. *Cardiovasc Res.*, 72(3):384–393, 2006.

[9] CG Broyden. The convergence of a class of double rank minimization algorithms. 2. the new algorithm. *Journal of the Institute of Mathematics and its Applications*, 6:222–231, 1970.

[10] AJ Butte, P Tamayo, D Slonim, TR Golub, und IS Kohane. Discovering functional relationships between rna expression and chemotherapeutic susceptibility using relevance networks. *Proc. Natl. Acad. Sci. U.S.A.*, 97:12182–12186, 2000.

[11] CM Carvalho, H Massam, und M West. Simulation of hyper-inverse wishart distributions on graphical models. *Biometrika*, 2007.

[12] R Castelo und A Roverato. A robust procedure for gaussian graphical model search from microarray data with p larger than n. *Journal of Machine Learning Research*, 7:2621–2650, 2006.

[13] RG Cowell, AP Dawid, SL Lauritzen, und DJ Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.

[14] JD Cox, S Barber-Derus, AJ Hartz, M Fischer, RW Byhardt, R Komaki, JF Wilson, und M Greenberg. Is adenocarcinoma/large cell carcinoma the most radiocurable type of cancer of the lung? *Int J Radiat Oncol Biol Phys.*, 12(10):1801–1805, 1986.

[15] CJ Creighton, AM Hilger, S Murthy, JM Rae, AM Chinnaiyan, und D El-Ashry. Activation of mitogen-activated protein kinase in estrogen receptor alpha-positive breast cancer cells in vitro induces an in vivo molecular phenotype of estrogen receptor alpha-negative human breast tumors. *Cancer Res.*, 66(7):3903–3911, 2006.

[16] AP Dawid und SL Lauritzen. Hyper markov laws in the statistical analysis of decomposable graphical models. *The Annals of Statistics*, 22(3), 1993.

[17] R Diestel. *Graphentheorie*. Springer, Berlin Heidelberg New York, 2000.

[18] Adrian Dobra, B Jones, C Hand, J Nevins, und M West. Sparse graphical models for exploring gene expression data. *Journal of multivariate Analysis*, 90:196–212, 2004.

[19] RL Dykstra. Establishing the positive definiteness of the sample covariance matrix. *Ann. Math. Statist.*, 41(6):2153–2154, 1970.

[20] D Edwards. *Introduction to Graphical Modelling*. Springer, 1995.

[21] B Efron. Large-scale simultaneous hypothesis testing: the choice of a null hypothesis. *J. Amer. Statist. Assoc.*, 99:96–104, 2004.

[22] P Erdös. Graph theory and probability. *Cannad. J. Math.*, 11:34–38, 1959.

[23] G Fischer. *Lineare Algebra*. Vieweg, Braunschweig/Wiesbaden, 1995.

[24] R Fletcher. A new approach to variable metric algorithms. *Computer Journal*, 13:317–322, 1970.

[25] E Fosslien. Review: Mitochondrial medicine–cardiomyopathy caused by defective oxidative phosphorylation. *Ann Clin Lab Sci.*, 33(4):371–395, 2003.

[26] ME Garber, OG Troyanskaya, K Schluens, S Petersen, Z Thaesler, M Pacyna-Gengelbach, M van de Rijn, GD Rosen, CM Perou, RI Whyte, RB Altman, PO Brown, D Botstein, und I Petersen. Diversity of gene expression in adenocarcinoma of the lung. *Proc. Natl. Acad. Sci. U.S.A.*, 98(24):13784–13789, 2001.

[27] R Gentleman und V Carey. Bioconductor. *R News*, 2(1):11–16, 2002.

[28] R Gentleman, E Whalen, W Huber, und S Falcon. *graph: A package to handle graph data structures*. R package version 1.12.1.

[29] S Gerschgorin. über die abgrenzung der eigenwerte einer matrix. *Izv. Akad. Nauk. USSR Otd. Fiz.-Mat. Nauk*, 7:749–754, 1931.

[30] D Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24:23–26, 1970.

[31] EC Goldsmith und TK Borg. The dynamic interaction of the extracellular matrix in cardiac remodeling. *J Card Fail.*, 8(6 Suppl):314–318, 2002.

[32] S Gruvberger, M Ringner, Y Chen, S Panavally, LH Saal, A Borg, M Fernö, C Peterson, und PS Meltzer. Estrogen receptor status in breast cancer is associated with remarkably distinct gene expression patterns. *Cancer Research*, 61:5979–5984, 2001.

[33] S Hannenhalli, ME Putt, JM Gilmore, J Wang, MS Parmacek, JA Epstein, EE Morrisey, KB Margulies, und TP Cappola. Transcriptional genomics associates fox transcription factors with human heart failure. *Circulation*, 114(12):1269–1276, 2006.

[34] E Herpel, M Pritsch, A Koch, TJ Dengler, P Schirmacher, und PA Schnabel. Interstitial fibrosis in the heart: differences in extracellular matrix proteins and matrix metalloproteinases in end-stage dilated, ischaemic and valvular cardiomyopathy. *Histopathology*, 48(6):736–747, 2006.

[35] NJ Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra Appl.*, 103:103–118, 1988.

[36] H Hotelling. New light on the correlation coefficient and its transform. *Journal of the Royal Statistical Society, Series B*, 15:193–232, 1953.

[37] R Ihaka und R Gentleman. R: a language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5:299–314, 1996.

[38] N Joza, GY Oudit, D Brown, P Bénit, Z Kassiri, N Vahsen, L Benoit, MM Patel, K Nowikovsky, A Vassault, PH Backx, T Wada, G Kroemer, P Rustin, und JM Penninger. Muscle-specific loss of apoptosis-inducing factor leads to mitochondrial dysfunction, skeletal muscle atrophy, and dilated cardiomyopathy. *Mol Cell Biol.*, 25(23):10261–10272, 2005.

[39] MM Kittleson, SQ Ye, RA Irizarry, KM Minhas, G Edness, JV Conte, G Parmigiani, LW Miller, Y Chen, JL Hall, JG Garcia, und JM Hare. Identification of a gene expression profile that differentiates between ischemic and nonischemic cardiomyopathy. *Circulation*, 110:3444–3451, 2004.

[40] SL Lauritzen. *Graphical Models*. Springer, 1996.

[41] O Ledoit und M Wolf. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *J. Empir. Finance*, 10:603–621, 2003.

[42] F Leisch. Dynamic generation of statistical reports using literate data analysis. *Compstat 2002 - Proceedings in Computational Statistics*, pages 575–580, 2002.

[43] TY Liu, CW Lin, S Falcon, J Zhang, und JW MacDonald. *hgu133a: Affymetrix Human Genome U133 Set Annotation Data (hgu133a)*. R package version 1.14.0.

[44] TY Liu, CW Lin, S Falcon, J Zhang, und JW MacDonald. *hgu95av2: Affymetrix Human Genome U95 Set Annotation Data (hgu95av2)*. R package version 1.14.0.

[45] TY Liu, CW Lin, S Falcon, J Zhang, und JW MacDonald. *hu6800: Affymetrix HuGeneFL Genome Array Annotation Data (hu6800)*. R package version 1.14.0.

[46] TY Liu, CW Lin, S Falcon, J Zhang, und JW MacDonald. *KEGG: A data package containing annotation data for KEGG*. R package version 1.14.1.

[47] D Madigan, SA Andersson, MD Perlman, und CT Volinsky. Bayesian model averaging and model selection for markov equivalence classes of acyclic digraphs. *Commumications in Statistics: Theory and Methods*, 25:2493–2520, 1996.

[48] J Marin-Garcia, Y Pi, und MJ Goldenthal. Mitochondrial-nuclear cross-talk in the aging and failing heart. *Cardiovasc Drugs Ther.*, 20(6):477–491, 2006.

[49] F Markowetz, J Bloch, und R Spang. Non-transcriptional pathway features reconstructed from secondary effects of rna interference. *Bioinformatics*, 21(21):4026–4032, 2005.

[50] N Meinshausen und P Bühlmann. High dimensional graphs and variable selection with the lasso. The Annals of Statistics, 2006.

[51] JA Nelder und R Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1964.

[52] MF Oliver, E Samuel, P Morley, GB Young, und PL Kapur. Detection of coronary-artery calcification during life. *Lancet*, 25(1):891–895, 1964.

[53] KB Reddy und S Glaros. Inhibition of the map kinase activity suppresses estrogen-induced breast tumor growth both in vitro and in vivo. *Int J Oncol.*, 30(4):971–975, 2007.

[54] F Rodriguez, F Perán, F Garrido, und F Ruiz-Cabello. Upmodulation by estrogen of hla class i expression in breast tumor cell lines. *Immunogenetics*, 39(3):161–167, 1994.

[55] A Roverato. Cholesky decomposition of a hyper inverse wishart matrix. *Biometrika*, 87(1):99–112, 2000.

[56] A Roverato. Hyper inverse wishart distribution for non-decomposable graphs and its application to bayesian inference for gaussian graphical models. *Scand. J. Statistics*, 29(3):391–411, 2002.

[57] H Rue und L Held. *Gaussian Markov Random Fields*. Chapman & Hall/CRC, 2005.

[58] Harvard Rue. Fast sampling of gaussian markov random fields. *J.R. Statist. Soc. B*, 63(2):325–338, 2001.

[59] M Ruschhaupt, W Huber, A Poustka, und U Mansmann. A compendium to ensure computational reproducibility in high-dimensional classification tasks. *Statist. Appl. Genet. Mol. Biol.*, 3.1(37), 2004.

[60] M Satoh, M Nakamura, T Akatsu, Y Shimoda, I Segawa, und K Hiramori. Toll-like receptor 4 is expressed with enteroviral replication in myocardium from patients with dilated cardiomyopathy. *Lab Invest.*, 84(2):173–181, 2004.

[61] M Satoh, G Tamura, und I Segawa. Enteroviral rna is endomyocardial biopsy tissues of myocarditis and dilated cardiomyopathy. *Pathol. Int.*, 44:345–351, 1994.

[62] G Sawitzki. Software components and document integration for statistical computing. *Proceedings ISI Helsinki 1999 (52nd session) Bulletin of the International Statistical Institute*, Tome LVIII(Book 2):117–120, 1999.

[63] J Schäfer und K Strimmer. An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6), 2005.

[64] J Schäfer und K Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statist. Appl. Genet. Mol. Biol.*, 4(32), 2006.

[65] J Schneider, M Ruschhaupt, A Buneß, M Asslaber, K Zatloukal, A Poustka, und H Sültmann. Identification and meta-analysis of a small gene expression signature for the diagnosis of estrogen receptor status in invasive ductal breast cancer. *Int. J. Canc.*, 119(12):2974–2979, 2006.

[66] DF Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24:647–656, 1970.

[67] M Steenman, G Lamirault, MN Le, CM Le, D Escande, und JJ Leger. Distinct molecular portraits of human failing hearts identified by dedicated cdna microarrays. *Eur J Heart Fail*, 7:157–165, 2005.

[68] AE Teschendorff, A Miremadi, SE Pinder, IO Ellis, und C Caldas. An immune response gene expression module identifies a good prognosis subtype in estrogen receptor negative breast cancer. *Genome Biology*, 8(8), 2007.

[69] *The Chipping Forecast*, volume 21, 1999.

[70] R Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

[71] R Tibshirani, T Hastie, B Narasimhan, und G Chu. Class prediction by nearest shrunken centroids, with application to dna microarrays. *Statistical Science*, 18:104–117, 2003.

[72] V Tusher, R Tibshirani, und G Chu. Significance analysis of microarrays applied to the ionizing radiation response. *PNAS*, 98:5116–5121, 2001.

[73] BF Uretsky, K Thygesen, PW Armstrong, JG Cleland, JD Horowitz, BM Massie, M Packer, PA Poole-Wilson, und L Ryden. Acute coronary findings at autopsy in heart failure patients with sudden death: results from the assessment of treatment with lisinopril and survival (atlas) trial. *Circulation*, 102:611–616, 2000.

[74] TH Wang, QL Xiang, JW Chen, H Pan H, und YH Cui. Raloxifene plus 17beta-estradiol inhibits proliferation of primary cultured vascular smooth muscle cells and human mammary endothelial cells via the janus kinase/signal transducer and activator of transcription3 cascade. *European Journal of Pharmacology*, 561(1-3):7–13, 2007.

[75] Y Wang, JGM Klijn, Y Zhang, AM Sieuwerts, MP Look, F Yang, D Talantov, M Timmermans, ME Meijer van Gelder, J Yu, T Jatkoe, EMJJ Berns, D Atkins, und JA Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365:671–679, 2005.

[76] M West, C Blanchette, H Dressman, E Huang, S Ishida, R Spang, H Zuzan, JA Jr Olson, JR Marks, und JR Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proc. Natl. Acad. Sci. U.S.A.*, 98(20):11462–11467, 2001.

[77] J Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley, New York, 1990.

[78] F Yang, JA Foekens, J Yu, AM Sieuwerts, M Timmermans, JG Klijn, D Atkins, Y Wang, und Y Jiang. Laser microdissection and microarray analysis of breast tumors reveal er-alpha related genes and pathways. *Oncogene*, 25(9):1413–1419, 2006.

Index

Danksagung

Ich möchte mich an dieser Stelle bei den Menschen bedanken, die mich bei der Fertigstellung dieser Arbeit unterstützt haben. Mein großer Dank gilt Prof. Ulrich Mansmann, der mich über die ganzen Jahre hinweg betreut hat. Durch ihn hab ich eine tiefes Verständniss für statistische Analysen bekommen, ohne das ich diese Arbeit niemals hätte fertigstellen können. Auch die Diskussionen waren immer sehr anregend und haben mir geholfen so manchen Zusammenhang zu verstehen. Zudem hat er mich ermutig, 2006 die Sommerschule in Saint Flour zu besuchen. Dies war sehr hilfreich, da ich dort auch die Möglichkeit hatte, mit Prof. Steffen Lauritzen zu diskutieren, dem 'Erfinder' der Graphenmoralisierung. Neben Prof. Mansmann danke ich in München auch allen anderen Mitarbeitern am IBE, die mich immer freundlich aufgenommen habe, wenn ich dort gearbeitet habe, und mir mit Rat und Tat zur Seite gestanden haben.

Danken möchte ich Prof. Jörg Rahnenführer von der Technischen Universität Dortmund dafür, dass er sich bereit erklärt hat, als zweiter Gutachter dieser Arbeit zu fungieren.

Auch danke ich den Mitarbeitern der Abteilung Molekulare Genomanalyse am Deutschen Krebsforschungszentrum Heidelberg unter der Leitung von Prof. Annemarie Poustka für die sehr gute und produktive Arbeitsathmosphäre in den ganzen Jahren. Insbesondere den Mitarbeitern und ehemaligen Mitarbeitern der 'Expression Profiling'-Gruppe unter der Leitung von PD Dr. Holger Sültmann gilt mein Dank. Ich habe hier nicht nur viel über die Microarray-Technologie und die damit verbundenen Analysen gelernt, sondern auch über die zu Grunde liegenden biologischen Zusammenhänge. Hier sei besonders Dr. Ruprecht Kuner erwähnt, der mir bei Fragestellungen in dieser Arbeit bezüglich Herzerkrankungen zur Seite gestanden hat.

Meine Arbeit am DKFZ und an der LMU wurde finanziell unterstützt vom Nationalen Genomforschungsnetz(NGFN: 01GR0459, 01GR0418).

Für die Unterstützung in all den Jahren danke ich meinen Eltern, meinem Bruder und meinen Freunden. Mein größter Dank gilt aber meiner Freundin Claudia Justus, und das nicht nur für das Korrekturlesen dieser Arbeit. Claudia hat mich immer wieder unterstützt und aufgebaut, wenn es mit der Arbeit mal etwas holprig oder schleppend vorangegangen ist, und durch sie hab ich auch gelernt, dass bei graphentheoretischen Zusammenhängen die Intuition zwar sehr wichtig ist, in den Beweisen aber nichts zu suchen hat. Danke dafür und für alles andere.

Lebenslauf

Name:	Markus Ruschhaupt
Anschrift:	Gundolfstraße 1 69120 Heidelberg
Geburtsdaten:	14. Juli 1975 in Werther (Westf.)

Schulbildung

1986 - 1995	Max-Planck-Gymnasium Bielefeld Abitur (Durchschnittsnote: 1,6)
1984 - 1988	Grundschule Werther

Studium

Okt. 1995 - Nov. 2002	Studium Diplom-Mathematik an der Universität Bielefeld
Nov. 2002	Diplom (Gesamtnote: Sehr gut)
Aug. 1997	Vordiplom (Gesamtnote: Sehr gut)

Beschäftigungen

Seit Februar 2005	Wissenschaftlicher Mitarbeiter an der Ludwig-Maximilians-Universität, Institut für medizinische Informatik, Biometrie und Epidemiologie, München
Seit Juni 2003	Wissenschaftlicher Mitarbeiter am Deutschen Krebsforschungszentrum, Abteilung für Molekulare Genomanalyse, Heidelberg
1998 - 2002	Tutor/Studentische Hilfskraft an der Fakultät für Mathematik, Bielefeld