Graphenrekonstruktion anhand abhängiger Zeitreihen in biologischen Netzwerken

TECHNISCHE UNIVERSITÄT DARMSTADT

vom Fachbereich Biologie der Technischen Universität Darmstadt

zur Erlangung des Grades Doctor rerum naturalium (Dr. rer. nat.)

Dissertation

von Patrick Boba

Erstgutachter: Prof. Dr. Kay Hamacher

Zweitgutachter: Prof. Dr. Gerhard Thiel

Darmstadt 2019

Boba, Patrick: Graphenrekonstruktion anhand abhängiger Zeitreihen in biologischen Netzwerken
Darmstadt, Technische Universität Darmstadt,
Jahr der Veröffentlichung der Dissertation auf Tuprints:
URN: urn:nbn:de:tuda-tuprints-91284
Tag der mündlichen Prüfung: 05.09.2019

I Zusammenfassung

Die Biologie befasst sich mit dem Aufbau und der Organisation von Lebewesen. Bei beiden Aspekten finden sich auf verschiedenen Abstraktionsebenen Phänomene, die sich als Netzwerke interpretieren lassen. Ein makroskopisches Beispiel dafür sind Räuber-Beute-Beziehungen (z. B. Größe einer Fuchspopulation in Abhängigkeit von ihren Beutetieren wie Kaninchen, Hühnern, etc.). Es ist leicht ersichtlich, dass die Größen der Populationen jeweils voneinander abhängen und eine wechselseitige Dynamik widerspiegeln. Auf molekularer Ebene gibt es ebenfalls Beispiele für Interaktionen, die sich über ein dynamisches Netzwerk beschreiben lassen, etwa bei zellulären Prozessen. Ein Beispiel hierfür ist die Katalyse einer chemischen Reaktion mittels eines Enzyms. Die Konzentration des Enzyms und der beteiligten Substanzen beeinflussen dabei die Geschwindigkeit, mit welcher der Stoffwechselprozess abläuft. Mit dieser (makro)molekularen Ebene beschäftigt sich diese Arbeit.

Wie wichtig ein funktionierendes Netzwerk ist, wird deutlich wenn man ein gestörtes System betrachtet, etwa wenn eingeschleppte Arten ein Ökosystem aus dem Gleichgewicht bringen. Ein aktuelles Beispiel dazu ist der amerikanische Kalikokrebs (*Orconectes immunis*), der sich derzeit in Europa schnell ausbreitet, da ihm natürliche Feinde fehlen. Gleichzeitig bedroht er durch seinen Ressourcenverbrauch Tierarten wie Libellen, Amphibien und einheimische Krebse.

Auf zellulärer Ebene kann eine Störung des Netzwerks der DNA-Reparatur und der Zellzykluskontrolle zu der Entstehung von Krebs führen. Die DNA-Reparatur stellt ein komplexes System aus verschiedenen Proteinen und DNA dar. Der Ausfall eines Bestandteils dieses Systems kann für den Reparaturprozess verheerende Folgen haben.

Es wird deutlich wie wichtig das Verständnis der Dynamik dieser Systeme ist, um Analysen und Prognosen für den Zustand dieser Systeme zu erstellen. In den beiden genannten Beispielen kann es helfen die Entstehung von Krebs besser vorherzusagen, bzw. bedrohte Tier- und Pflanzenarten zu schützen.

Anhand von Netzwerken, die die Interaktion von Proteinen, DNA und RNA darstellen, ist das Ziel dieser Arbeit, den messbaren Informationsfluss zwischen verschiedenen beteiligten Elementen zu erkennen und mit dessen Hilfe die Struktur des Netzwerks zu rekonstruieren. Zu diesem Zweck werden die Zeitreihen der einzelnen Knoten mittels verschiedener statistischer und informationstheoretischer Maße miteinander in Beziehung gesetzt.

Bei der Auswahl der verschiedenen Maße greife ich sowohl auf klassische statistische Maße (z. B. Korrelationskoeffizienten), als auch auf informationstheoretische (auf Shannon-Entropie basierende) Methoden zurück, die in den letzten Jahren im Bereich der Biologie populärer gewordenen sind.

Der Vergleich dieser Methoden findet durch mehrere Beispielsysteme statt, die ich in

drei verschiedene Kategorien eingeteilt habe. Allen Beispielen gemein ist die zeitliche Simulation, um ein dynamisches, veränderliches System abzubilden. Mit Hilfe der Messung des Zusammenhangs der einzelnen Knoten über die Zeit, soll im Umkehrschluss auf die Topologie des zugrunde liegenden Netzwerks zurück geschlossen werden.

In die erste Kategorie fällt ein einfaches Differentialgleichungssystem, welches zwei Feedback-Schleifen miteinander koppelt. Die Parametrisierung des Netzwerks sorgt für eine stabile Schwingung der beiden Schleifen um ihren jeweiligen Mittelwert.

Als nächste Kategorie werden zwei verschiedene Typen von Zufallsgraphen erzeugt. Der erste wird durch einem von mir entworfenen Algorithmus erstellt, der eine bestimmte Menge an Knoten erzeugt, die mit einer bestimmten Anzahl von Eingangskanten und Ausgangskanten verbunden sind. Der zweite Typus ist ein sogenanntes skalenfreies Netz. Diese Netzwerktopologie kann in vielen Systemen wieder gefunden werden. Dazu gehören sowohl biologische als auch auch digitale soziale Netzwerke.

In der letzten Kategorie wende ich die genannten Methoden auf verschiedene Beispiele aus der BioModels Database an. Diese Datenbank bietet sich aufgrund der umfangreichen Datensätze an und enthält viele biochemische Netzwerke, z. B. Protein-Protein-Interaktion, Protein-RNA-Interaktion usw.

Abschließend diskutiere ich die vorgelegten Ergebnisse und gebe einen Ausblick auf die Möglichkeiten diese Ansätze weiter zu verfolgen und auszubauen. Des Weiteren wurden im Zuge dieser Arbeit verschiedene Software Tools von mir entwickelt, bzw. studentische Arbeiten zur Entwicklung betreut, die für die Durchführung der hier gezeigten Analysen wichtig waren. Diese werden in einem getrennten Abschnitt besprochen.

II Summary

Biology deals with the construction and organization of living things. For both aspects phenomena can be found on different abstraction levels that can be interpreted as networks. A macroscopic example of this is predator-prey relationships (e.g. the size of a fox population depending on its prey, such as rabbits, chickens, etc.). It is easy to see that the sizes of these populations depend on each other and reflect a mutual dynamic. At the molecular level, there are also examples of interactions that can be described via a dynamic network, such as cellular processes. An example is the catalysis of a chemical reaction by means of an enzyme. The concentrations of the enzyme and substances involved influence the speed with which the metabolic process takes place. With this (macro) molecular level deals this thesis.

How important a functioning network is, becomes apparent when analyzing disturbed system. This is the case when foreign species are introduced into an ecosystem. A recent example of this is the American *Orconectes immunis*, a crayfish which is currently spreading fast throughout Europe since its lack of natural predators. At the same time it threatens local animal species such as dragonflies, amphibians and native Crustacean by competing for resources.

At the cellular level, a disorder of the DNA repair network and the cell cycle control can lead to the development of cancer. DNA repair is a complex system that involves various proteins and DNA. A failure of one of the components of this system can have devastating consequences for the repair process.

It becomes clear how important it is to understand the dynamics of these systems: Creating predictions for the state and dynamics of these systems will in case of the former save endangered species or the latter help to fight cancer.

The goal of this work is to reconstruct network topologies by measuring the information flow between involved elements of the network like proteins, DNA or RNA. For this purpose the time series of each node (e.g. a protein) is put in relationship to the other nodes using various statistical and information-theoretical measures.

The selection of reviewed methods consists of both classical statistical measures (e.g. correlation coefficients) and measures that became popular more recently in the field of biology like information-theoretical methods based on Shannon entropy.

The comparison of these methods is based on several example models. All these systems in common is the temporal simulation, to depict a dynamically changing system. Based on the measurement of the relationship between the single nodes over time, the idea is to inversely refer to the topology of the underlying network.

The most simple one of those examples is a system of differential equations, which couples two feedback loops. The parameterization of the network caused a stable oscillation of the two loops around their respective mean value.

The next category contains two different types of random graphs. The first type is created by an algorithm I have designed in a way that it randomly generates an amount of nodes with a certain in- and outdegree without individual nodes being isolated. The second type is a so-called scale-free network. This network topology can be found in many types of systems, such as biological networks or social networks.

As a final step, I apply the above mentioned methods to various examples from the Biomodels Database. It contains a large amount of biochemical datasets, e.g. networks of protein-protein interaction, protein-RNA interaction, etc.

Finally, I discuss the results presented and give an outlook of the opportunities to pursue and further develop these approaches. Additionally I describe various software tools that I either developed in the course of this work, or supervised the development by student projects. These tools were crucial to carrying out the analyzes shown in this thesis and are discussed in an individual section.

III Publikationen

Teile dieser Arbeit basieren auf den folgenden Publikationen:

- Patrick Boba, Stefan Gries, Kay Hamacher (2014) R as an Integration Tool in High Performance Computing – Lessons Learned. *Lecture Notes in Informatics (LNI)*

  Das Manuskript wurde von mir angefertigt. Die Betreuung des Studenten, der die Entwicklung des R-Pakets durchführte, fand durch mich statt. Das Paper wurde von mir im Rahmen eines Vortrags an der INFORMATIK 2014 (Gesellschaft für Informatik - GI) vorgestellt.

- Patrick Boba, Dominik Bollmann, Daniel Schoepe, Nora Wester, Jan Wiesel und Kay Hamacher (2015) Efficient computation and statistical assessment of transfer entropy. *Frontiers in Physics*

  Das Manuskript wurde von mir angefertigt. Die Betreuung der Studenten, die die C++ Bibliothek entwickelten, wurde von mir durchgeführt.

- Patrick Boba und Kay Hamacher (2017) TransferEntropyPT: An R Package to Assess Transfer Entropies via Permutation Tests. *Springer, Lecture Notes in Computer Science*

  Das Manuskript wurde in Zusammenarbeit mit K. Hamacher angefertigt. Das zugrunde liegende R-Paket wurde von mir entwickelt.

Inhaltsverzeichnis

1 Einleitung

1.1 Systembiologie

Systembiologie beschreibt einen Zweig der Biologie, der sich mit dem Verständnis der Organismen als Ganzes beschäftigt [94]. Während zum Beispiel die Genetik sich primär auf Prozesse im Zusammenhang mit DNA und Genregulation konzentriert, beschreibt die Proteomik die Erforschung der zu einem bestimmten Zeitpunkt interagierenden Proteine. Die Systembiologie versucht einzelne Teilbereiche wie Genetik und Proteomik mit zeitlichen und räumlichen Abläufen zu integrieren, um ein möglichst realistisches Gesamtbild eines Lebewesens zu erzeugen. Das grundlegende Verständnis dieser Systeme und die Interaktion deren Komponenten ist seit Jahrzehnten Feld intensiver Forschung. Dies spiegelt sich auch in dem 2017 vergebenen Nobelpreis für Medizin und Physiologie wieder. Dieser wurde an Jeffrey C. Hall, Michael Rosbash und Michael W. Young für die Erforschung des zirkadianen Rhythmus vergeben [54, 124, 29]. Dieser beschreibt die Fähigkeit biologischer Organismen, physiologische Vorgänge in regelmäßigen Perioden zu organisieren. Dies verdeutlicht einen weiteren wichtigen Aspekt der Systembiologie, nämlich die Zeit bzw. den Zeitpunkt als Faktor für den Zustand eines biologischen Systems.

Biologische Systeme können auf unterschiedlichen Organisationsebenen betrachtet werden. Die Spannweite reicht von Systemen auf Populationsebene im makroskopischen bis zur Interaktion von Molekülen oder Atomen im mikroskopischen Bereich. Das Ziel der Systembiologie ist es möglichst exakte (mathematische) Modelle für die einzelnen Ebenen zu entwickeln oder in Multiskalenansätzen die verschiedenen Komplexitätsebenen zu verbinden [84]. Um Aussagen über ein System treffen zu können, verwendet man Modelle, die möglichst exakt das reale System, bzw. dessen betrachtete Eigenschaften abbilden. Beispielsweise kann man es als Netzwerkgraph notieren, um Erkenntnisse über die Topologie des Systems zu erlangen. So kann man beispielsweise zeigen, ob ob es Pfade zwischen allen Knoten gibt, oder es zentrale Knoten gibt, die von besonders vielen anderen Knoten aus erreicht werden können. Abschnitt 3.4.1 gibt einen genaueren Überblick zum Thema Graphen.

Eine andere Perspektive auf das gleiche System bekommt man, indem man dessen Dynamik untersucht. Dabei geht es um die Änderung des Zustands eines Knotens in Abhängigkeit von der Zeit. Es wir aber nicht nur ein einzelner Knoten betrachtet, sondern auch dessen verbundene Nachbarknoten. Diese Modellierung kann durch ein System von Differentialgleichungen (siehe Abschnitt 2.5) geschehen, um zum Beispiel die Änderung der Konzentration eines Metaboliten in einem Stoffwechselkreislauf zu beschreiben, bzw. vorherzusagen.

Die Modellierung eines Systems und dessen Simulation erfolgt am Computer mittels numerischer Methoden. Netzwerksimulationen spielen heute eine große Rolle in den Biowissenschaften [76] und darüber hinaus [7]. Während in den Anfängen der Systembiologie zunächst nur vereinfachte Modelle erstellt werden konnten, ermöglichte die steigende Rechenleistung [86, 112] und die verbesserte Datengrundlage das Aufbauen immer komplexerer Systeme. Besonders in der Systembiologie sind Simulationen metabolischer Netzwerke zu einer *State-of-the-art*-Methode zur Analyse von Eigenschaften und Dynamik geworden und weisen den experimentell bestimmten Netzwerktopologien evolutionäre Bedeutung zu [33]. Hier stellen sich bioinformatischen Ansätzen aber gleich mehrere Herausforderungen, die häufig in diesem Forschungszweig anzutreffen sind: Legacy-Code-Infrastruktur, mehrere oft widersprüchliche Programmierparadigmen und schlecht kuratierte Datenquellen. Letzteres stellte ein Problem bei der Auswahl der biologischen Modelle in Abschnitt 3.5 dar, sodass manuelle Eingriffe in Modelle und Software erforderlich waren. Für hypothesengeleitete Untersuchungsansätze im Bereich der Systembiologie werden häufig hohe Anforderungen an HPC-Ressourcen gestellt. Gleichzeitig sind visuelle/interaktive Analysen erforderlich. In meiner Publikation [16] stelle ich daher eine Lösung vor, die beide Ansätze kombiniert und auch in dieser Arbeit zum Einsatz kommt (siehe Abschnitt 4.1).

Mit der gestiegenen Rechenleistung ging auch die Notwendigkeit einher, größere Datenmengen zu organisieren und bereitzustellen. Dies führte zur Entstehung mehrerer großer Datenbanken, die in der Regel über das Internet erreichbar sind. Das US-amerikanische National Center of Biotechnology (NCBI) betreibt in verschiedenen Sparten der Biologie Datenbanken. Die bekanntesten sind vermutlich die Genbank [11] und die Protein Data Bank [12]. Das europäische Pendant dazu ist das European Bioinformatics Institute (EMBL-EBI). Hier findet man unter anderem die BioModels Database [75]. Als Datenformat wird dabei die Systems Biology Markup Language (SBML) [39] verwendet. Dieses auf XML basierende Format wurde entwickelt, um biochemische Netzwerke darzustellen und wird von vielen Programmen für Systembiologie unterstützt (Details in Abschnitt 4).

Ein Anwendungsbereich der Systembiologie ist unter anderem die Medikamentenforschung [13]. Da es immer mehr Hinweise darauf gibt, dass Wirkstoffe nicht nur mit einem spezifischen Ziel interagieren sondern oft mit mehreren, rückt die Analyse komplexer Netzwerke und das Verständnis auf Systemebene stärker in den Vordergrund [95]. Interessant ist dabei die Frage, ob es in diesen Netzwerken nach dem Prinzip eines „Generalschlüssels“ zu Interaktionen mit Makromolekülen kommt. Die Idee ist dabei, dass eine Substanz nicht nur mit einem Molekül des Netzwerks interagiert, um einen Effekt hervorzurufen sondern gleich mit mehreren Schlüsselstellen dieses Systems [83].

1.2 Netzwerke

Netzwerke bzw. Netzwerkmodelle finden in vielen wissenschaftlichen Domänen Anwendung. Der Wahl des richtigen Modells kommt dabei eine entscheidende Bedeutung zu [24]. Ein Aspekt ist dabei die Topologie des Graphen, d. h. wie ist die Anordnung und

Verbindung der einzelnen Knoten zueinander. Ein anderer Aspekt ist die Dynamik. Dabei spielt es eine Rolle, wie zeitliche Änderungen des Netzwerks beschrieben werden. Im Falle biologischer Netzwerke werden dafür zur Modellierung häufig Differentialgleichungen verwendet. Die Dynamik jedes Knotens wird dann mithilfe einer Gleichung beschrieben und die Änderung seines Zustands ergibt sich aus dem Lösen der Gleichung für einen bestimmten Zeitpunkt.

In dieser Arbeit liegt jedoch der Fokus auf biologischen, intrazellulären Netzwerke und einige daran angelehnte Modellsysteme, die im nächsten Abschnitt näher erläutert werden.

1.2.1 Biologische Netzwerke

Makromolekulare Netzwerke lassen sich grob in verschiedene Kategorien einteilen. Zwar sind natürlich keine klaren Grenzen zwischen den einzelnen Typen gezogen – Das Gegenteil ist eigentlich der Grundgedanke der Systembiologie: Nämlich das System als interagierendes Ganzes zu verstehen und nicht nur als einzelne Komponenten – aber die Definitionen helfen in der Auswahl der Analysemethoden und fokussieren sich üblicherweise auf eine bestimmte Fragestellung [28]. Die nachstehende Liste nennt die derzeit in der molekularen Systembiologie üblichen Einteilung von Netzwerken:

- Chemische Komponenten
- Proteinstrukturen
- Protein-Protein-Interaktionen
- Signalnetzwerke
- Gen-Interaktionsnetzwerke
- Metabolische Netzwerke

Die direkte und wahrscheinlich einfachste Methode chemische Reaktionen darzustellen, ist sie als Reaktionsgraphen abzubilden. Diese Notation erlaubt einen qualitativen Überblick über die Beziehung der interagierenden chemischen Entitäten. Diese vereinfachte Darstellung kann nun um quantitative Eigenschaften wie Reaktionskinetik und Konzentrationen der Reaktanden erweitert werden. Kombiniert man nun mehrere Reaktionsschritte erhält man ein Netzwerk an Reaktionen. Im Falle von Stoffwechselreaktionen spricht man dann von einem metabolischen Netzwerk.

Eine einfache Beschreibung einer enzymatischen Reaktion ist in Gleichung 1.1 zu sehen. Im ersten Schritt lagern sich Enzym und Substrat aneinander. Die Reaktionskonstanten k_1 und k_{-1} beschreiben die Geschwindigkeit mit der Substrat und Enzym sich anlagern, bzw. dissoziieren, da dieser Schritt reversibel ist. Über den Enzym-Substrat-Komplex ES wird anschließend das Produkt P per katalytischer Reaktion k_2 erzeugt [60].

$$\mathrm{E} + \mathrm{S} \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} \mathrm{ES} \xrightarrow{k_2} \mathrm{E} + \mathrm{P} \tag{1.1}$$

Ein Netzwerk aus solchen voneinander abhängigen Reaktionen kann allgemein als Differentialgleichungssystem (Abschnitt 2.5.2) beschrieben werden. Mittels numerischer Lösung (siehe 2.5.4) kann ein zeitlicher Verlauf der Reaktionen und somit der Konzentrationen der einzelnen Reaktanden simuliert werden. Dabei bedient man sich der Michaelis-Menten-Kinetik (Gleichung 1.2) die aus der Reaktionsgleichung hergeleitet werden kann [94].

$$v_0 = \frac{v_{\max} \cdot [S]}{K_m + [S]} \tag{1.2}$$

Die Reaktionsgeschwindigkeit v_0 zum Zeitpunkt $t = 0$ wird dabei von der maximalen Reaktionsgeschwindigkeit $v_{\max}$, der Substratkonzentration $[S]$ und der Michaelis-Menten-Konstante K_m bestimmt.

1.3 Kausalität

Die Frage nach dem ursächlichen Zusammenhang zwischen Objekten aller Art ist ein fundamentales Problem über verschiedene Wissenschaftszweige hinweg. Bereits die griechischen Philosophen [81] beschäftigten sich mit der Thematik; die Stoiker betrachteten Kausalität als universelles Prinzip [111].

Der naturwissenschaftliche Begriff der Kausalität ist, obwohl das Konzept allgemein leicht verständlich, nur schwer mathematisch klar zu definieren. Eine Herausforderung liegt dabei in der notwendigen zeitlichen Abfolge von Dingen, wie zwei oder mehrere Variablen. Das „Ursache-Wirkung-Prinzip“ ist eine zunächst anschauliche Formulierung des Konzepts der Kausalität: Ein kausaler Zusammenhang besteht zum Beispiel zwischen A und B, wenn B nur unter der Voraussetzung von A eintritt. Die ursprüngliche Idee von Wiener [121] wurde von Granger [47] aufgegriffen und formuliert (*Wiener-Granger Causality* oder kurz WGC): Es seien X und Y zwei Zeitreihen. Dann hängt Y kausal von X ab, wenn wir Y_{t+1} unter Beobachtung von X **und** Y signifikant zuverlässiger vorhersagen, als unter Beobachtung von Y allein [20].

Einen Überblick über verschiedene Methoden des Auffindens solcher kausalen Zusammenhänge in Zeitreihen wird zum Beispiel von Hlaváčková-Schindler et al. [56] gegeben.

1.4 Motivation: Rekonstruktion von Netzwerken

Das Ziel dieser Arbeit ist es, anhand von knotenbasierten Zeitreihen eines Netzwerks die Topologie, also den Zusammenhang der Knoten, dieses Netzwerks zu rekonstruieren. Mittels der verschiedenen eingesetzten Korrelationsmaße, die in Abschnitt 2 detailliert

beschrieben werden, soll festgestellt werden, ob eine Rekonstruktion auch bei geringer Anzahl von Datenpunkten, d. h. kurzen Zeitreihen, möglich ist. Dieser Ansatz wurde bereits in früheren Arbeiten mit jeweils unterschiedlichem Fokus der Analyse verfolgt [113].

Hauptsächlich wurden bei bisherigen Studien DNA/RNA-Microarrays als Quelle der Primärdaten verwendet. Van der Heijden et al. [114] verwendeten Bayes'sche Netzwerke um Verschlechterungen des Gesundheitszustandes von Patienten mit chronisch obstruktiver Lungenerkrankung (COPD) vorherzusagen. Die verwendete Methode hat jedoch den Nachteil, dass sie nur auf azyklische Graphen angewendet werden kann. Da viele biologische Netzwerke jedoch zyklische Elemente im Sinne von Feedback-Schleifen enthalten (so auch die hier verwendeten Netzwerke), findet diese Methode in dieser Arbeit keine Anwendung.

Um Proteinkonzentrationen zu schätzen wurden bis vor wenigen Jahren üblicherweise Konzentrationen von mRNA mittels microarrays gemessen. Aus diesen wurde die Konzentration der Proteine anhand ihrer zugehörigen mRNA (englisch *messenger RNA*) abgeleitet. Der Nachteil dieser indirekten Methode ist, dass man dabei Effekte wie die tatsächliche Geschwindigkeit der Proteinsynthese sowie den Abbau oder die generelle Lebensdauer eines Proteins außer Acht lässt [115]. Daher haben sich mittlerweile weitere Verfahren wie die Orbitrap Massenspektrometrie etabliert, die die direkte Messung von Konzentrationen erlauben [45]. Ein verbreiteter Ansatz nutzt Störungsereignisse des Netzwerks („Network Inference“) der Zeitreihen, um die Struktur herzuleiten [107]. Dazu werden einzelne Knoten des Netzwerks entfernt. Es können so relevante Pfade in einem Netzwerk identifiziert werden, zum Beispiel weil ein stabiler Zustand des Netzwerks damit zerstört wird. Es können aber auch Redundanzen kenntlich gemacht werden, wenn das Netzwerk weiterhin funktioniert, da andere Pfade zur Kompensation des fehlenden Knoten verwendet werden.

Um eine ausreichende Datengrundlage zu schaffen, war es zunächst notwendig eine Plattform zu finden, mit der die Simulation solcher Netzwerke effizient (auch mit großen Datenmengen) stattfinden kann. Die Statistiksprache `R` stellt solch eine Plattform dar [97]. Sie ist vielseitig, leistungsfähig und wird auch in „Big-Data-Anwendungen“ und „Data-Mining“ eingesetzt (z. B. genomische Datenanalyse [14, 18, 89]).

Ein weiterer Vorteil von `R` ist die Möglichkeit dessen Funktionalität durch Pakete zu erweitern. Zum einen kann man dadurch auf bereits vorhandene Funktionen von Drittanbietern zurückgreifen, zum anderen ist es möglich eigene Pakete zu entwickeln, die gegebenenfalls fehlende Funktionalität zur Verfügung stellen können. Die Erweiterungen beschränken sich nicht nur auf Code, der in `R` selbst geschrieben ist. Vielmehr können Low-Level-Sprachen wie z. B. C++ eingebunden werden, was häufig eine deutlich höhere Performance in der Ausführung bietet, da diese hardwarenahe Programmierung erlauben (siehe vor allem das RCpp-Paket [35]).

Aus den genannten Anforderungen an die Methoden ergeben sich zwei Hauptbestandteile hinsichtlich der computergestützten Auswertung. Der Erste ist das Erzeugen der Zeitreihen anhand bekannter Netzwerkstrukturen. Der Zweite ist das Berechnen, bzw. Bewerten des Informationsaustauschs zwischen den Zeitreihen dieser Netzwerke. Für beide Bestandteile wurden hier `R`-Pakete entwickelt.

Das erste Paket beinhaltet einen numerischen Solver für stochastische Differentialgleichungen und ist in der Lage XML-Daten aus der BioModels Database zu importieren. Da die Pakete nicht nur für den internen Gebrauch bestimmt waren, war an dieser Stelle eine der größten Herausforderungen die effiziente und benutzerfreundliche Integration verschiedener externer Tools und Datenquellen, ohne dabei den Nutzer in seinen gewünschten Analyseverfahren einzuschränken. Ein weiterer Aspekt war die Wiederverwertbarkeit: Das System sollte in der Lage sein, möglichst jedes Netzwerk der BioModels Database zu berechnen.

Der detaillierte Aufbau und die Funktionsweise des ersten Pakets (SOMNIBIEN) wird in Abschnitt 4.1, sowie in meiner Publikationen [16] beschrieben. Als Basis dieses Paketes dienten zwei externe Softwarelösungen, die das Konvertieren der Netzwerke aus ihrem nativen SBML-Dateiformat und das numerische Lösen der stochastischen Differentialgleichungen ermöglichten. Diese wurden um weitere Code-Bausteine ergänzt und anschließend als `R`-Pakets verfügbar gemacht.

Um auch die Analyse der Daten mittels der entropiebasierten Maße zu ermöglichen, wurde zum einen das Paket BioPyhsConnectoR [58] genutzt und zum anderen ein weiteres `R`-Pakets entwickelt. Dieses dient der effizienten Berechnung der Transferentropie (TE) [15] inklusive eines Permutationstests zur Feststellung der statistischen Signifikanz (siehe Abschnitt 4.3). Hier machen wir von der Möglichkeit Gebrauch, mittels der Einbindung von C++ Code eine hohe Rechenleistung zur erzielen, ohne auf den Komfort für die Auswertung durch `R` zu verzichten. Das Paket TransferEntropyPT [17] bietet zu diesem Zweck effiziente statistische Signifikanztests ergänzend zur Berechnung der Transferentropie.

2 Material und Methoden

Die Analyse der Zeitreihen in dieser Arbeit und Rekonstruktion der Netzwerke erfolgte mittels verschiedener statistischer und informationstheoretischer Maße. Informationstheorie beschäftigt sich mit dem Übertragen von Information, sowie deren Quantifizierung und Speicherung. Eins der wichtigsten Maße in diesem Zusammenhang stellt die von Claude E. Shannon definierte Entropie dar [103]. In Abschnitt 2.1 dieses Kapitels werden zunächst die verwendeten varianzbasierten (z. B. Korrelationskoeffizienten) und im folgenden Abschnitt 2.2 entropiebasierte Maße (z. B. Transinformation) erläutert.

Die verwendeten Datensätze sind Zeitreihen, die in Bezug auf ihren Informationstransfer miteinander untersucht werden sollen. Eine Zeitreihe ist eine Folge, deren Werte in diskreten Abständen vorliegen. Bei kontinuierlichen Signalen wird zum Erzeugen einer Zeitreihe daher die Quelle in definierten (häufig regelmäßigen) Abständen abgetastet. Ein stochastischer Prozess, der konsekutive diskrete Signale produziert, kann dazu dienen Zeitreihen zu erzeugen.

2.1 Varianzbasierte Maße

2.1.1 Erwartungswert und Varianz

Der Erwartungswert μ einer diskreten Zufallsvariablen X ist der Mittelwert aller, nach ihrer Wahrscheinlichkeit gewichteten, möglichen Ausgänge dieser Variablen, bzw. die Summe der Produkte aus der Realisierung eines Ereignisses x_i und deren Wahrscheinlichkeiten $p(x_i)$:

$$E[X] = \mu_X = \sum_{i=1}^{n} x_i \, p(x_i). \tag{2.1}$$

Der Schätzer des Erwartungswert $\hat{\mu}$ ist der Mittelwert aller Beobachtungen dieser Variablen.

Eine weitere wichtige Eigenschaft von Verteilungen ist die Varianz. Für Gauß-Verteilungen beschreibt die Varianz die Streuung der Verteilung. Für die diskrete Zufallsvariable X ist sie definiert als der Erwartungswert des quadratischen Abstands vom Mittelwert von X:

$$Var(X) = \sigma_X^2 = \sum_{i=1}^{n} (x_i - \mu_X)^2 \, p(x_i). \tag{2.2}$$

Die Wurzel der Varianz wird als Standardabweichung σ bezeichnet.

2.1.2 Kovarianz und Korrelation

Kovarianz

Die Kovarianz C misst lineare Abhängigkeiten zwischen zwei Variablen. Sie stellt die gemeinsame Varianz zweier Variablen um ihren jeweiligen Mittelwert dar.

Für Zeitreihen X und Y ist die Stichprobenkovarianz definiert als:

$$C(X,Y) = \frac{1}{t}\sum_{i=1}^{t}(x_i - \hat{\mu}_X)(y_i - \hat{\mu}_Y) \tag{2.3}$$

Dabei wird über t Werte die Differenz zum entsprechenden Mittelwert $\hat{\mu}$ gebildet. Positive Werte drücken einen gleichsinnigen und negative Werte einen gegensinnigen Zusammenhang aus. Ist die Kovarianz null, ist kein linearer Zusammenhang der Variablen feststellbar.

Die Kovarianz ist kein normiertes Maß. Daraus resultiert, dass ein Vergleich zwischen verschiedenen Kovarianzen nur möglich ist, wenn die Daten aus Experimenten stammen, die den gleichen Wertebereich und die gleiche Dimension haben. Daher wird in der Praxis üblicherweise ein auf der Kovarianz aufbauender Korrelationskoeffizient berechnet.

Pearsons Produkt-Moment-Korrelation

Die Pearson-Korrelation r ist ein auf der Kovarianz basierendes Maß um Abhängigkeiten zwischen zwei Variablen zu beschreiben [91]. Dabei ist σ die Standardabweichung der jeweiligen Variablen. Die Korrelation, definiert als

$$r_{X,Y} = \frac{C(X,Y)}{\sigma_X \sigma_Y}, \tag{2.4}$$

nimmt dadurch Werte im Bereich $[-1, 1]$ an. Dabei stellen die Grenzen einen maximalen linearen, bzw. gegensinnig linearen Zusammenhang dar. Ein Wert von Null bedeutet die Variablen sind unkorreliert.

Spearmans Rangkorrelationskoeffizient

Abgesehen von Pearsons r, dem vermutlich bekanntesten Korrelationsmaß, gibt es noch weitere, als Korrelation bezeichnete Maße. Der Spearman Rangkorrelationskoeffizient von [108] ist eine Erweiterung der Produkt-Moment-Korrelation. Anstatt der Variablen selbst werden deren Ränge R verwendet. Das Anordnen nach Rang ist eine Transformation der Daten, bei der die Werte der Variable X der Größe nach aufsteigend sortiert werden (der kleinste Wert bekommt den Rang eins, der nächst größere den Rang zwei, usw.). Anschließend werden anstatt der tatsächlichen Werte von X die Rangzahlen verwendet. Dadurch schwächt Spearmans ρ vor allem den Einfluss von Ausreißern auf die Berechnung ab und ist robuster gegenüber nichtlinearen Zusammenhängen. Außerdem

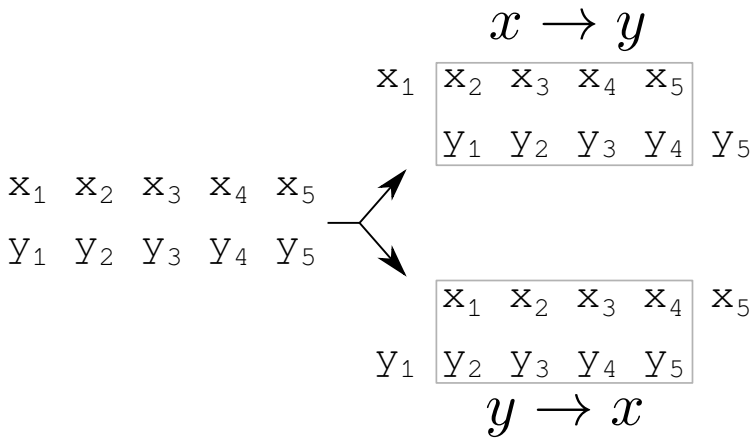


Abbildung 2.1: Die beiden Vektoren können in unterschiedliche Richtungen relativ zueinander verschoben werden. Dadurch bekommt man zwei neue Vektorpaare, die jeweils um zwei Elemente kürzer sind.

erhöht dies bei Kombination mehrerer Variablen mit unterschiedlicher Skala die Vergleichbarkeit untereinander.

Die Definition ist analog zu Gleichung 2.4, jedoch mit den Rängen $R(X)$ und $R(Y)$ der beiden Variablen:

$$\rho_{X,Y} = \frac{C(R(X), R(Y))}{\sigma_{R(X)}\sigma_{R(Y)}}. \tag{2.5}$$

Ein Nachteil bei der Verwendung von Rängen der Variablen ist das Auftreten von gleichen Werten in den Variablen. Die Berechnung des Rangs in einem solchen Fall kann auf verschiedene Weisen gelöst werden („tie breaking“ [62]).

Beide in diesem Abschnitt genannten Korrelationskoeffizienten (Gleichungen 2.4 und 2.5) berücksichtigen nicht den möglichen zeitlichen Versatz zwischen zwei Vektoren, bzw. Zeitreihen. Daher wurden die Datenvektoren zur Analyse in beide möglichen Richtungen um eine (oder mehrere) Position verschoben (zu sehen in Abb. 2.1). Daraus ergibt sich jedoch eine verkürzte Sequenz, die zum Berechnen verwendet wird.

2.1.3 Informationsflussrate (Rate of Information Flow)

Eine weitere Methode basierend auf der Kovarianz von Variablen ist die von [77] vorgestellte Informationsflussrate RIF. Ähnlich wie die Transferentropie in Abschnitt 2.2.5, versucht die RIF Zusammenhänge zwischen verschiedenen Zeitreihen herzustellen, indem ein zeitlicher Versatz der Variablen berücksichtigt wird.

$$RIF_{y\rightarrow x} = \frac{C_{x,x}C_{x,y}C_{y,\mathrm{d}\,x} - C_{x,y}^2 C_{x,\mathrm{d}\,x}}{C_{x,x}^2 C_{y,y} - C_{x,x}C_{x,y}^2} \tag{2.6}$$

Hier steht $C_{x,y}$ für die Kovarianz und $C_{y,\mathrm{d}x}$ für die Kovarianz der um Schrittweite d verschobenen Zeitreihe x und der unveränderten Zeitreihe y. Im Gegensatz zu den

bisher beschriebenen Methoden ist die RIF ein asymmetrisches Maß, sodass $RIF_{y \to x} \neq RIF_{x \to y}$ gilt.

2.1.4 Kreuzkorrelation

Ein aus der Signalverarbeitung stammendes Maß ist die Kreuzkorrelation, die im Englischen als Cross Correlation Function (CCF) bezeichnet wird [85]. Dabei wird eine der beiden untersuchten Zeitreihen um d Zeitschritte verschoben. Sie wird dann als sogenanntes „sliding dot product“ berechnet. Für den diskreten Fall ist die Kreuzkovarianz definiert als:

$$r_k(x, y) = \frac{c_k(x, y)}{\sqrt{c_0(x, x)c_0(y, y)}} \tag{2.7}$$

Darauf wird die Normierung für die als Stichprobenkreuzkorrelation wie folgt berechnet:

$$c_k(x, y) = \frac{1}{n} \sum_{t=1}^{n-d} (x_{t+d} - \hat{\mu}_X)(y_t - \hat{\mu}_Y) \tag{2.8}$$

Für jede Verschiebung d kann für jeden Zeitschritt t das Produkt des Werts x und y gebildet werden. Die Verschiebung d wird schrittweise angepasst und maximal die Länge T der Zeitreihen erreicht. Ein Spezialfall der CCF ist die Autokorrelation (Autocorrelation Function (ACF)), bei der $y = x$ gilt. Diese zielt darauf ab Periodizität innerhalb einer Zeitreihe festzustellen.

2.2 Entropiebasierte Maße

Im Gegensatz zu den im vorigen Abschnitt vorgestellten varianzbasierten Maßen, beruhen die nachfolgenden Methoden auf dem Konzept der Shannon-Entropie [103]. Diese und weitere darauf fußende Konzepte werden im folgenden Abschnitt besprochen. Ein Vorteil gegenüber varianzbasierten Maßen ist die Möglichkeit auch Variablen zu untersuchen, die nicht-reelle Zahlenwerte (wie etwa Klassenlabels) annehmen. Mit ihnen ist es außerdem leichter möglich, nicht-lineare Zusammenhänge zwischen Variablen zu erfassen.

2.2.1 Informationsgehalt

Als Basis der weiteren Methoden dient der Informationsgehalt. Dieser gibt an, wie viel Information eine Nachricht (Variable) enthält. Er ist abhängig von der Mächtigkeit des Alphabets (die möglichen Zustände, die eine Variable annehmen kann) und den Häufigkeiten ihres Auftretens. Hat eine Signalquelle eine Mächtigkeit von fünf, kann diese

Quelle eine Nachricht mit potentiell fünf verschiedenen Symbolen aussenden. Nicht notwendigerweise entspricht die Anzahl an übertragenen Symbolen einer Signalquelle auch dem Informationsgehalt, da sie Redundanzen enthalten kann.

Der mathematische Begriff der Information wurde von Claude E. Shannon [103] definiert als der negative Logarithmus der Wahrscheinlichkeit $p(x)$ des Auftretens von Symbol x.

$$I(x) = -\log_2 p(x) \tag{2.9}$$

Im Allgemeinen hat sich als Einheit für die Information das *bit* durchgesetzt. Damit wird der Informationsgehalt über den Logarithmus zur Basis 2 definiert. Die in dieser Arbeit verwendeten Maße basieren immer auf dieser Basis, sofern nicht explizit anders angegeben.

2.2.2 Shannon-Entropie

Als Ausgangspunkt für Untersuchungen zur Informationsübertragung dient in vielen wissenschaftlichen Bereichen die sogenannte Shannon-Entropie [103]. Sie ist definiert als der mittlere Informationsgehalt (oder auch Erwartungswert der Information $E(I)$) eines Signals. Dies kann zum Beispiel eine Textnachricht oder auch eine Gensequenz sein. In dieser Arbeit soll damit der Zusammenhang und Informationsfluss zwischen Zeitreihen analysiert werden. In der folgenden Form ist die Entropie für eine beliebige diskrete Variable X auf dem endlichen Alphabet $A = \{x_1, x_2, ..., x_n\}$ definiert als:

$$H(X) = -\sum_{x \in A} p(x) \log_2 p(x) \tag{2.10}$$

Es sei $p(x)$ die Wahrscheinlichkeit (relative Häufigkeit) eines Symbols x. Die Obergrenze der Entropie (und somit auch die Obergrenzen der darauf basierenden Maße) ist abhängig von der Alphabetgröße. Das Maximum der Entropie wird bei einer Gleichverteilung aller Werte erreicht. Die Untergrenze der Entropie ist null, was einer Variablen mit konstantem Wert entsprechen würde.

2.2.3 Kullback-Leibler-Divergenz

Auf der Formulierung der Entropie basierend gibt es verschiedene Ansätze Signale miteinander zu vergleichen. Die Kullback-Leibler-Divergenz (D_{KL}) [72] stellt sich als Unterschied der Entropie zweier Verteilungen dar und wurde bereits vielfach zu Analysen im biologischen Bereich eingesetzt [51, 82]. Die D_{KL} ist somit asymmetrisch definiert für die Verteilungen $P(x)$ und $Q(x)$:

$$D_{KL}(P||Q) = \sum_x P(x) \ \log_2 \frac{P(x)}{Q(x)} \tag{2.11}$$

2.2.4 Transinformation (Mutual Information)

Die Transinformation (im Folgenden MI) stellt ein auf der Shannon-Entropie beruhendes informationstheoretisches Maß und eine Sonderform der D_{KL} dar [37, 46]. Dabei wird der Unterschied einer gemeinsamen Verteilung zweier Variablen P zur unabhängigen Verteilung $Q = p(x)\,p(y)$ dieser beiden Variablen gemessen. Die MI ist im Gegensatz zur D_{KL} jedoch aufgrund ihrer Konstruktion symmetrisch.

$$MI(X,Y) = \sum_{x,y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)} \tag{2.12}$$

Formuliert als Summe der Entropien ist die MI:

$$MI(X,Y) = H(X) + H(Y) - H(X,Y) \tag{2.13}$$

Die MI kann nur Werte ≥ 0 annehmen. Ihre Obergrenze hängt von den Einzelentropien $H(X)$ und $H(Y)$ ab. Die gemeinsame Entropie der Variablen $H(X,Y)$ ist mindestens so hoch wie die größte Entropie von X oder Y, sodass gilt:

$$H(X,Y) \geq \max\big(H(X), H(Y)\big) \tag{2.14}$$

Mit Gleichung 2.13 folgt daraus:

$$\max\big(MI(X,Y)\big) \leq \min\big(H(X), H(Y)\big) \tag{2.15}$$

Wie eingangs in 2.2.2 erwähnt, ist die Entropie maximal bei einer Gleichverteilung der Variablen. Nimmt man eine Alphabetgröße von 16 an, ergibt sich somit für die Einzelentropie eine maximale Obergrenze von $-\log_2 \frac{1}{16} = 4\,\text{bit}$). Die theoretische Obergrenze der gemeinsamen Entropie ist im Extremfall dann gleich der Einzelentropie (entspricht einer eindeutigen Zuordnung der beiden Variablen). In diesem Fall wäre die Information, die X über Y enthält, maximal und im genannten Beispiel MI $= 4\,\text{bit}$. Ein Wert von null wiederum würde bedeuten das keine Information über Y enthalten ist.

Die MI hat bereits in vielen wissenschaftlichen Feldern Anwendung gefunden, darunter Neurobiologie [105], Biophysik [68], Protein-Co-Evolution [18] und viele weitere. Ein Vorteil dieser Methode gegenüber kovarianzbasierten Maßen ist das Erfassen von nichtlinearen Abhängigkeiten. Das Konzept der MI ist auch potentiell erweiterbar auf mehr als zwei Variablen, wie unter anderem von Waechter et al. [117] gezeigt.

Zeitverzögerte Transinformation (Time Delayed Mutual Information)

Jedoch stellt die MI in ihrer ursprünglichen Form nur eine bedingt geeignete Methode zur Analyse von Zeitreihen dar, da die Chronologie der Realisierungen keine Berücksichtigung findet. Als Erweiterung wurde daher eine zeitverzögerte MI (TDMI) vorgestellt

[41] und unter anderem zur Analyse von Elektroenzephalografie-(EEG)-Zeitreihen angewendet [65]. Die zeitverzögerte MI stellt eine Modifikation von Gleichung 2.12 dar. Dabei wird anstelle des gleichen Zeitpunks für y ein um d verschobener Wert gewählt (Abb. 2.2a). Dadurch wird der Zusammenhang zwischen zwei Variablen nicht zum gleichen Zeitpunkt analysiert sondern mit einem zeitlichen Versatz.

$$\mathrm{TDMI}(X_n \to Y_{n+d}) = \sum_{x,y} p(x_n, y_{n+d}) \log \frac{p(x_n, y_{n+d})}{p(x_n)p(y_{n+d})} \tag{2.16}$$

Im Gegensatz zur normalen MI gilt für $\mathrm{TDMI}(X_n \to Y_{n+d}) \neq \mathrm{TDMI}(Y_n \to X_{n+d})$. Sie ist also asymmetrisch in Bezug auf die Variablen.

2.2.5 Transferentropie

Dieses Konzept fortführend entwickelte [102] die Transferentropie. Zunächst betrachten wir dazu die Entropierate H_r eines Prozesses $\{x_t\}$ für diskrete, äquidistante Zeitschritte t.

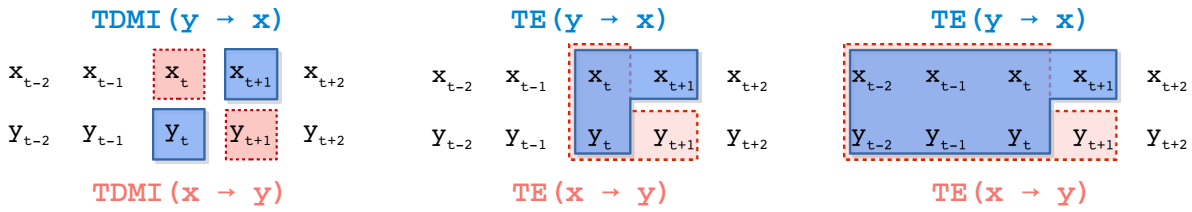
$$H_r = -\sum_{x_{t+1}, x_t^l} p(x_{t+1}, x_t^l)\ \log_2 p(x_{t+1} \mid x_t^l) \tag{2.17}$$

Dabei stellt x_t^l ein l-Tupel der Stichprobe zum Zeitpunkt $t, t-1, \ldots, t-l+1$ dar. Die bedingte Wahrscheinlichkeit $p(x_{t+1}|x_t^l)$ gibt die Wahrscheinlichkeit eines Auftretens von x_{t+1} unter vorangegangenem Auftreten von x_t^l an. Wie sich der Formel bereits entnehmen lässt, steigt mit größerem l auch die Anzahl der möglichen Tupel exponentiell und somit der Rechenaufwand für das Erstellen der Histogramme der Wahrscheinlichkeiten entsprechend schnell an.

Einem ähnlichen Ansatz wie dem der TDMI folgend wird nun die Transferentropie analog zur D_{KL} formuliert. Anstatt nur Werte zum Zeitpunkt t zu betrachten, werden dabei auch vorangegangene Werte der jeweiligen Zeitreihen nach einem bestimmten Schema betrachtet. Die Formel stellt sich, ebenso wie die MI, als D_{KL} dar:

$$\mathrm{TE}\,(Y \to X) = \sum p(x_{t+1}, x_t^l, y_t^m)\ \log_2 \frac{p(x_{t+1} \mid x_t^l, y_t^m)}{p(x_{t+1} \mid x_t^l)} \tag{2.18}$$

Die Variablen X und Y sind Zeitreihen auf einem diskreten Alphabet. Zu jedem Zeitpunkt $t+1$ für x wird außerdem der Wert von x_t und y_t ermittelt. Die Parameter l und m sind die Reichweite in die „Vergangenheit“ der Vektoren für x, respektive y (im Folgenden als *Fenster* bezeichnet). Für die Parameter l und m können beliebige ganzzahlige positive Werte eingesetzt werden. Entsprechend der Definition für den Prozess $\{y_t\}$ kann l eine andere Zeitfensterlänge als die für $\{x_t\}$ mit $l \neq m$ annehmen. Für den simpelsten Fall gilt somit $l = m = 1$, was dem Schema in Abbildung 2.2b entspricht. In dieser setze ich immer $m = l$, sodass sich Formel 2.19 zu



(a) Time Delayed MI (b) TE mit Fenster $l = 1$ (c) TE mit Fenster $l = 3$

Abbildung 2.2: Vergleich zwischen den Konzepten der (a) TDMI und der TE mit Fenster (b) $l = 1$ und (c) $l = 3$. Die Farben geben an welche Richtung der TDMI und TE berechnet werden: rot $(x \to y)$, blau $(y \to x)$

$$\mathrm{TE}\,(Y \to X) = \sum p(x_{t+1}, x_t^l, y_t^l)\ \log_2 \frac{p(x_{t+1} \mid x_t^l, y_t^l)}{p(x_{t+1} \mid x_t^l)} \tag{2.19}$$

umformulieren lässt.

Für die TE gilt, ebenso wie für die TDMI, dass sie asymmetrisch in der Berechnung für zwei Variablen X und Y ist, sodass:

$$\mathrm{TE}(Y \to X)\ \neq\ \mathrm{TE}(X \to Y). \tag{2.20}$$

Die Transferentropie wurde in den letzten Jahren bereits häufig und erfolgreich eingesetzt. Vor allem in den Neurowissenschaften erfreut sie sich großer Beliebtheit [120, 63]. Aber auch in anderen Feldern, wie Wettervorhersage [70] oder Wirtschaftswissenschaften [6], findet sie Anwendung. Für gaußverteilte Variablen konnte gezeigt werden, dass die Transferentropie und Granger-Causality äquivalent sind [9].

Schranken und Normierung der Transferentropie

Um die Ergebnisse der TE besser miteinander vergleichbar zu machen, nehme ich in dieser Arbeit eine Normierung auf das (approximierte) Minimum und Maximum der TE vor. Während die theoretische untere Schranke der TE bei null liegt, hängt die obere Schranke direkt von der Anzahl der möglichen Realisierungen in den Einträgen der beiden Vektoren x und y ab. Die TE bekommt dadurch feste Grenzen und ist nicht mehr von den tatsächlichen Alphabeten der beiden Vektoren abhängig. Dadurch lassen sich auch verschiedene Zeitreihen-Paare miteinander direkt vergleichen, sofern die gewählten Parameter (Fenstergrößen l und m) der TE gleich sind. Die normierte TE ergibt sich dabei wie folgt:

$$\widehat{\mathrm{TE}} = \frac{\mathrm{TE} - \min(\mathrm{TE})}{\max(\mathrm{TE}) - \min(\mathrm{TE})} \tag{2.21}$$

Da bis heute keine analytische Lösung bereit steht, verwendete ich zum Approximieren der jeweiligen Minima und Maxima einen sogenannten Greedy-Algorithmus. Die Implementierung der Software wird in Abschnitt 4.3.1 im Detail beschrieben.

2.3 Diskretisierung (Binning)

In der Praxis wird zur Berechnung der in Abschnitt 2.2 genannten entropiebasierten Definitionen eine Diskretisierung der Daten durchgeführt. Bei dieser Datenreduktion wird eine Klasseneinteilung der Variablen, im Folgenden „Binning“ genannt, vorgenommen. Sie ist für alle nachfolgend beschriebenen Methoden von großer Bedeutung und wird daher in diesem Abschnitt kurz beschrieben.

Messdaten in biologischen Experimenten sind häufig kontinuierliche Werte. Bei der Analyse mit informationstheoretischen Methoden werden üblicherweise Häufigkeiten für Ereignisse (etwa die Häufigkeit des Auftreten eines bestimmten Messwerts) betrachtet. Bei kontinuierlichen Daten (besonders bei geringer Stichprobengröße) ist nun die Wahrscheinlichkeit hoch, dass jedes Ereignis nur einmal auftritt. Die daraus erzeugten Histogramme der Einzelwahrscheinlichkeiten sind deswegen dünn besetzt, woraus sich wiederum das Problem ergibt, dass mithilfe entropiebasierter Methoden ein kausaler Zusammenhang zwischen den Variablen schlecht nachgewiesen werden kann.

Um diesem Problem entgegen zu wirken und aus solchen Daten dennoch Erkenntnisse abzuleiten, wird deswegen ein Binning durchgeführt. Die hier verwendete Methode teilt die Wertebereiche der kontinuierlichen Daten in gleich große Abschnitte, genannt „Bins“, ein (siehe 2.3). Dadurch werden die Daten in ein diskretes „Alphabet“ überführt und können mit den in Abschnitt 2.2 entropiebasierten Methoden ausgewertet werden.

Die ideale Größe der Bins, bzw. deren Anzahl ist allerdings nicht einfach zu definieren. Eine höhere Anzahl stellt zwar potentiell eine bessere Auflösung der Ursprungsdaten dar, kann aber auch wieder bei einer zu hohen Wahl zur Problematik der dünn besetzten Histogramme führen. Ein größerer Stichprobenumfang würde dem zwar entgegen wirken, allerdings steht dieser bei Experimenten aus dem Wet-Lab häufig nicht zur Verfügung. Für die Transferentropie konnte gezeigt werden, dass erst ab einer bestimmten Bingröße, der Informationsfluss korrekt erkannt wird [50]. Diese Information wurde als Anhaltspunkt für die Auswahl in den nachfolgenden Untersuchungen berücksichtigt.

Ein weiterer Vorteil des Binnings ist es, den Effekt von statistischem Rauschen der Daten abzumildern, da die Maße nicht Bezug auf den exakten Wert nehmen, sondern vielmehr auf die Einteilung des Wertes in eine Kategorie. Mit dem Binning verbunden ist allerdings das Risiko, dass Information verloren geht, da nach dem Binning keine Unterscheidung zwischen den Datenpunkten innerhalb eines Bins gemacht werden kann. Damit kann man das Histogramm für die Einteilung der Werte definieren als:

$$h(i) = \frac{1}{L} \sum_{k=1}^{L} \mathbb{1}_{[\tilde{x}_i \leq x_k < \tilde{x}_{i+1}]} \tag{2.22}$$

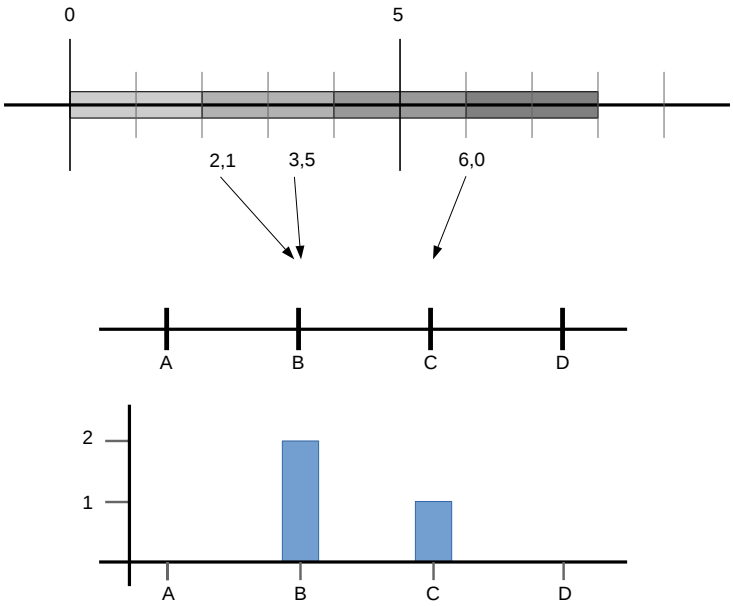


Abbildung 2.3: Die Abbildung veranschaulicht das Konzept der Diskretisierung. Die tatsächlichen Messwerte sind in dem Beispiel drei reelle Zahlen im Bereich zwischen null und acht (oben). Die Realisierungen werden anhand des Binnings auf vier diskrete Bins (A bis D) abgebildet (Mitte) und anschließend gezählt. Daraus resultiert das Histogramm (unten).

Die Kategorisierung der Werte in Bins $h(i)$ erfolgt über äquidistante Grenzen der Bins $\{\tilde{x}_1, \ldots, \tilde{x}_n\}$ für die einzelnen Werte x_k der Reihe $\{x_1, \ldots x_n\}$.

2.4 Statistische Signifikanz

2.4.1 p-Wert (p-value)

Um für die Ergebnisse der in Abschnitt 2.1.2 beschriebenen Korrelationskoeffizienten einen Konfidenzbereich anzugeben, wurde der p-Wert als Signifikanzmaß verwendet [59]. Der p-Wert (oft auch Signifikanzwert) beschreibt einen einseitigen Hypothesentest, der prüft mit welcher Wahrscheinlichkeit das erhaltene Ergebnis unter Annahme einer Nullhypothese zufällig erzeugt werden kann. Als Grenze für ein statistisch signifikantes Ergebnis wird ein Wert von $0,05$ angenommen. Dieser Wert entspricht ungefähr dem zweifachen der Standardabweichung, der üblicherweise in wissenschaftlichen Studien verwendet wird [80].

2.4.2 Z-Scores

Man kann zur Feststellung der statistischen Signifikanz einen sogenannten Z-Test durchführen und erhält daraus einen Z-Score (auch Z-Transformation) [55]. Für den Test wer-

den die ursprünglichen Zeitreihen randomisiert. Dadurch wird die Abfolge der Zeitreihe zwar zerstört (die Reihenfolge der Symbole x einer Zeitreihe wird verändert), der Inhalt selbst bleibt aber bestehen. Auf Basis der randomisierten Daten werden dann die entsprechenden Maße erneut berechnet. Der Vorgang des Randomisierens und Berechnens wird mehrfach durchgeführt und anschließend Mittelwert und Standardabweichung berechnet. Diese Werte verwendet der Z-Score um eine Abweichung des Ergebnisses vom Mittelwert der randomisierten Verteilung in Einheiten der Standardabweichung zu erhalten:

$$\mathrm{Z} = \frac{x - \hat{\mu}_x}{\hat{\sigma}_x} \tag{2.23}$$

Hierbei sei x die Stichprobe, sowie $\hat{\mu}_x$ und $\hat{\sigma}_x$ Schätzer des Mittelwerts und Schätzer der Standardabweichung der randomisierten Verteilung. Beispielsweise, für ein Ergebnis mit einem Z-Score von drei, befindet sich das tatsächliche Ergebnis den dreifachen Wert der Standardabweichungen entfernt von dem Mittelwert einer zufälligen Verteilung des Ergebnisses.

2.4.3 Qualität der Messmethoden

Um die Qualität von Analysemethoden zu beurteilen wird häufig deren Grenzwertoptimierungskurve, im Englischen als Receiver-Operating-Characteristic (ROC) bezeichnet, berechnet. Sie setzt die Effizienz mit der Fehlerrate für verschiedene Parametersätze ins Verhältnis. Die Auswertung einer solchen Kurve wird in der Regel anhand der Area under Curve (AUC). Dabei wird die Fläche unterhalb der Kurve berechnet. Um bestimmte Sonderfälle bei den Auswertungen in dieser Arbeit zu berücksichtigen, habe ich außerdem das Konzept um einen Korrekturfaktor erweitert. Dadurch ergibt sich dann als Maß das Volumen unterhalb einer Fläche: Volumen under Surface (VUS). Die einzelnen Konzepte werden in den nächsten drei Abschnitten genauer erläutert.

Receiver-Operating-Characteristic (ROC)

Bei einem Zufallsexperiment, dass einen binären Ausgang hat (positiv und negativ), kann eine Analysemethode den Ausgang vorhersagen. Daraus ergeben sich die vier Möglichkeiten wie in der Kontingenztabelle in Abbildung 2.4 gezeigt. Sagt die Methode ein positives Ergebnis vorher, welches korrekt ist, spricht man von True Positive (TP). Ist dies eine falsche Vorhersage wird es als False Positive (FP) bezeichnet. Analog dazu werden korrekt und falsch vorhergesagten negativen Ergebnisse True Negative (TN), bzw. False Negative (FN) genannt.

Area under Curve (AUC)

Um die Qualität einer Messmethode zu beurteilen, vergleicht man die Rate der Richtig-Positiven, auch True Positive Rate (TPR) mit der Rate der Falsch-Positiven (FPR).

		Eingetretenes Ereignis	
		Ereignis positiv	*Ereignis negativ*
Vorhergesagtes Ereignis	*Vorhersage positiv*	Richtig-Positiv (True Positive = TP)	Falsch-Positiv (False Positive = FP)
	Vorhersage negativ	Falsch-Negativ (False Negative = FN)	Richtig-Negativ (True Negative = TN)

Abbildung 2.4: Kontingenztabelle der Kategorien der ROC. Die bewertete Methode kann die Zustände „positiv“ und „negativ“ vorhersagen. Anhand des tatsächlichen Ereignisses wird festgestellt, ob die Vorhersage richtig oder falsch war. Daraus ergeben sich die vier Kategorien „Richtig-Positiv“ (TP) „Falsch-Positiv“ (FP), „Falsch-Negativ“ (FN) und „Richtig-Negativ“ (TN).

Gegeneinander aufgetragen ergibt sich daraus die ROC-Kurve (Abbildung 2.5). Zur Berechnung dieses Verhältnisses verwendet man die sogenannte Area under Curve (AUC). Ergibt das Integral der Kurve eins, bedeutet es demnach eine vollständige Erkennung aller Positiven ohne dabei Falsch-Positive zu erzeugen.

Häufig wird zur besseren Darstellung die Diagonale dem Plot hinzugefügt. Je weiter die Werte über der Diagonalen liegen, desto besser ist das Verhältnis zwischen Spezifität und Sensitivität. Liegen die Werte unterhalb der Diagonalen, deutet es darauf hin, dass die Methode nicht geeignet ist, das Messergebnis richtig zu bewerten.

In einigen Fällen kommt es vor, dass die Berechnung der Maße (z. B. Z-Scores oder bei der Normierung der Transferentropie) als Wert `Inf` (Infinite, Unendlich) oder `NaN` (Not a Number, Keine Zahl) zurückgeben. Dies passiert, wenn Berechnungen sehr kleine oder große Zahlen erzeugen und bei einer Division diese danach gegen unendlich konvergieren (`Inf`). Bei einigen Ansätzen der Berechnung wurden außerdem Ergebnisse mit niedriger Signifikanz (z. B. Z-Score) aus den Ergebnissen entfernt. In dem nächsten Abschnitt stelle ich daher eine modifizierte Variante der AUC vor, die dies berücksichtigt.

Volumen under Surface

Das oben erwähnte Fehlen von Werten bei der Messung der Transferentropie, führt zu der Problematik, dass die AUC nicht korrekt berechnet werden kann. Diese Werte dürfen nicht mit null gleich gesetzt werden, da dadurch fälschlicherweise die Annahme getroffen würde, dass kein Informationsfluss statt findet. Daher habe ich in einem weiteren Schritt die berechnete AUC mit dem Anteil der tatsächlich berechneten Werte normiert. Sind beispielsweise von 40 Variablen nur 30 Werte reelle Zahlen und sonst `Inf` in einer Auswertung, wird der erhaltene AUC mit 0.75 multipliziert. Damit lässt sich der so erzeugte Wert als Volumen unter eine Fläche (**V**olume **U**nder **S**urface) auffassen, da eine weitere Dimension in die Berechnung aufgenommen wird.

$$\mathrm{VUS} = \mathrm{AUC}\,\frac{N_R}{N} \tag{2.24}$$

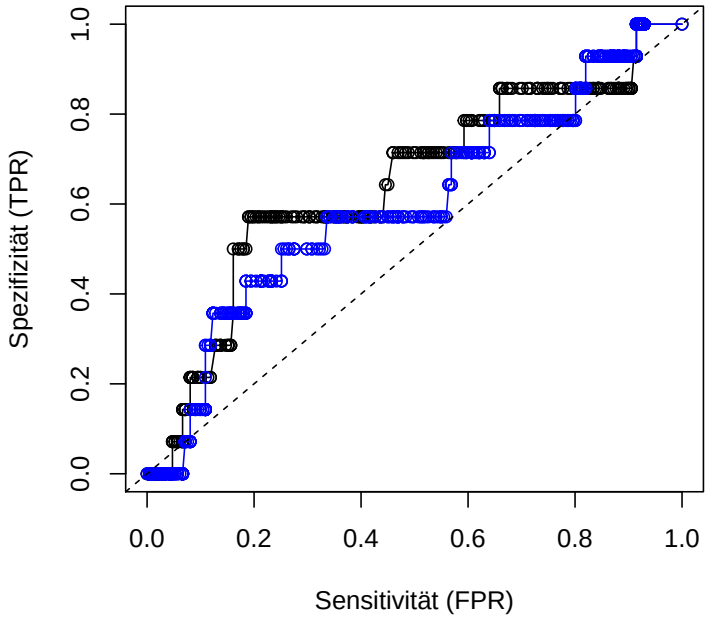


Abbildung 2.5: Beispiel einer Receiver-Operating-Characteristic-Kurve. Die durch die schwarze Linie repräsentierte Methode scheint in diesem Beispiel etwas bessere Vorhersageeigenschaften zu haben als die blaue.

Dabei ist N die Anzahl *aller* berechneten Ergebnisse, und N_R die Anzahl der verwendbaren Ergebnisse, nach Abzug aller `NA` oder `Inf`.

2.5 Differentialgleichungen

2.5.1 Gewöhnliche Differentialgleichungen

Eine Gewöhnliche Differentialgleichung (DGL), oder im Englischen *Ordinary Differential Equation (ODE)* ist eine Gleichung, die die Ableitung nach *einer* Variable enthält. Da viele physikalische Prozesse durch sie ausgedrückt werden, sind sie in den Naturwissenschaften sehr verbreitet. Oft, aber nicht ausschließlich, handelt es sich dabei um eine Ableitung nach der Zeit (z. B.: Beschleunigung ist die zweimalige Ableitung der Strecke nach der Zeit). Gegeben sei die chemische Reaktionsgleichung in 2.25:

$$A \xrightarrow{v_1} B \tag{2.25}$$

Diese chemische Reaktion kann als stöchiometrische Änderung der beteiligten Moleküle über die Zeit aufgeschrieben werden, sodass wir die Differentialgleichungen 2.26 erhalten.

$$\frac{\mathrm{d}A}{\mathrm{d}t} = -v_1 \quad \text{und} \quad \frac{\mathrm{d}B}{\mathrm{d}t} = v_1 \tag{2.26}$$

Die Gleichungen sagen nun aus, dass die Substanz A mit der gleichen Geschwindigkeit v_1 abgebaut wird, mit der die Substanz B aufgebaut wird.

Eine analytische Lösung solcher Gleichungen gestaltet sich häufig als schwierig, da sie weitaus komplexer sind als das oben gezeigte Beispiel. In diesem Fall werden numerische Algorithmen verwendet, um eine Approximation der Lösung zu erhalten. Einen kurzen Überblick der wichtigsten Algorithmen gibt Abschnitt 2.5.4.

2.5.2 Differentialgleichungssysteme

Sind mehrere Differentialgleichungen voneinander abhängig und lassen sich als System von Vektoren formulieren, spricht man von einem Differentialgleichungssystem. Man kann nun annehmen, dass in einem komplexen System von Reaktionen j mit der Reaktionsgeschwindigkeit v_j zu jeder Substanz M_i der stöchiometrische Koeffizient c_{ij} existiert, sodass man das Gleichungssystem wie folgt notieren kann:

$$\frac{\mathrm{d}M_i}{\mathrm{d}t} = \sum_{j=1}^{r} c_{ij} v_j(M_1, ..., M_r) \tag{2.27}$$

Ein einfaches Beispiel ist die aus der Ökologie stammende Lotka-Volterra-Gleichung [2]. Formel 2.28 modelliert den Zusammenhang einer Räuber-Beute-Beziehung. Die jeweiligen Populationen sind dabei als r (Räuber) und b (Beute) gekennzeichnet. Die zugehörigen Sterberaten S und Geburtenraten G komplettieren die DGL. Somit ist die Änderung der jeweiligen Population (r' und b') über die Zeit t in Abhängigkeit von Räuber- und Beutepopulation formuliert.

$$\begin{aligned} r'(t) &= -r(t)(S_r - G_r b(t)) \\ b'(t) &= b(t)(G_b - S_b r(t)) \end{aligned} \tag{2.28}$$

Bei der Lösung solcher DGL-Systeme unterscheidet man weiterhin zwei prinzipielle Typen. Bei gewöhnlichen Differentialgleichungen (also ODEs) findet die Ableitung nur nach einer Variablen statt. Ist zur Lösung der Gleichung die Ableitung nach mehreren Variablen erforderlich, handelt es sich um eine *partielle Differentialgleichung*. Um eine partielle DGL zu lösen, bedarf es jedoch korrekt formulierter Rand- oder Anfangsbedingungen. Die in dieser Arbeit verwendeten Systeme sind alle ODEs.

2.5.3 Stochastische Differentialgleichungen

Die vorigen Abschnitt beschriebenen Systeme haben als Grundannahme das sogenannte „Perfect Mixing“ (Perfekte Durchmischung). Es wird also davon ausgegangen, dass es

keine räumlichen Gradienten verschiedener Systemgrößen, wie Temperatur oder Konzentration gibt, es also ein *ideales* System ist. Um Modelle mit einem höheren Grad an Realismus zu erreichen (z. B. Umwelteinflüsse in die Räuber-Beute-Beziehung einfließen zu lassen), kann man das DGL-System aus 2.27 mit einem Rauschterm (Wiener-Prozess) erweitern. Diese wird dann als stochastischen Differentialgleichung oder SDE (aus dem englischen **S**tochastic **D**ifferential **E**quation) bezeichnet:

$$\mathrm{d}M_i(t) = f_i\left(M_1(t), \ldots, M_r(t), v_i^{\downarrow}, v_i^{\uparrow}\right)\mathrm{d}t + \beta \cdot M_i \cdot \mathrm{d}B_t^{(i)} \tag{2.29}$$

Hier sind $v_i^{\downarrow}, v_i^{\uparrow}$ die Geschwindigkeiten der Hin- und Rückreaktionen der Substanz i. Des Weiteren bestimmt der Faktor β die Stärke des Rauschens und $B_t^{(i)}$ beschreibt einen Wiener-Prozess. Man beachte, dass sowohl unabhängig und identisch verteiltes (independent and identically distributed = i.i.d.) Rauschen $B_t^{(i)} \neq B_t^{(j)}$ für $i \neq j$, als auch globales Rauschen mit $B_t^{(i)} := B_t$ in dieses Modell integriert sind. Ersteres simuliert Abweichungen vom Perfect Mixing, letzteres Temperaturänderungen des Systems (vgl. [44]). Details zur Konstruktion solcher Modelle sind auch zu finden in [53].

2.5.4 DGL-Lösungsverfahren

Es gibt verschiedene numerische Verfahren um Anfangswertprobleme bei DGLs approximativ zu lösen. Prinzipiell wird zwischen Einschritt- und Mehrschrittverfahren unterschieden [90]. Während erstere nur den aktuellen Schritt als Ausgang zur Lösung des nächsten Zeitpunkts verwenden, benutzen Mehrschrittverfahren auch vorherige Schritte. Die nächsten Abschnitte stellen das einfache Euler-Verfahren und das (wahrscheinlich am meisten benutzte) Runge-Kutta-Verfahren vor.

Euler-Methode

Die Euler-Methode (auch Euler-Verfahren) ist das einfachste und älteste Verfahren zum Lösen gewöhnlicher Differentialgleichungen [98]. Es ist ein Verfahren erster Konvergenzordnung, was bedeutet, dass der Fehler pro Zeitschritt i proportional zum Quadrat der Schrittweite ist. Diese Eigenschaft macht die Euler-Methode bei kleinen Schritten weniger genau als das im nächsten Abschnitt beschriebene Runge-Kutta-Verfahren.

Der generelle Ablauf der Methode ist wie folgt: Die Kurve wird schrittweise durch einzelne Punkte P_i approximiert. Unter der Voraussetzung, dass es bereits einen bekannten Startpunkt $P_0[t_0, y_0]$ auf der sonst noch unbekannten Kurve gibt, kann mit Hilfe der Differentialgleichung die Tangente berechnet werden, auf der wiederum der nächste Punkte $P_1[t_1, y_1]$ liegt.

Für ein Anfangswertproblem mit bekanntem $y(t_0) = y_0$ lässt sich auf diskreten Abschnitten t_i die Lösung der Gleichung

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} = y'(t) = f(t, y(t)) \tag{2.30}$$

approximieren, indem man sie wie folgt in eine Geradengleichung mit der Steigung ϕ zum Zeitpunkt t_i umformuliert:

$$y_{i+1} = y_i + \phi_i h \qquad \phi_i = f(t_i, y_i) \tag{2.31}$$

Je kleiner man die Schrittweite h bei diesem Verfahren wählt, desto genauer wird die tatsächliche Funktion approximiert, aber ebenso steigt der Rechenaufwand für den kompletten Zeitraum T.

Runge-Kutta-Methode (RK)

Da die Bezeichnung Runge-Kutta (RK)-Verfahren eine Familie verschiedener Verfahren erfasst, wird an dieser Stelle beispielhaft das oft verwendete Runge-Kutta-Verfahren der Ordnung 4 erläutert. Die Schrittweite h wird nun, entsprechend der Ordnung des Verfahrens, in vier Teilschritte k_n zerlegt, sodass sich für ϕh ergibt:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{2.32}$$

Die Werte k_n berechnen sich rekursiv.

$$\begin{aligned} k_1 &= f(t_i, y_i) \\ k_2 &= f(t_i + \frac{h}{2}, y_i + \frac{h}{2}k_1) \\ k_3 &= f(t_i + \frac{h}{2}, y_i + \frac{h}{2}k_2) \\ k_4 &= f(t_i + h, y_i + hk_3) \end{aligned} \tag{2.33}$$

Zu erkennen ist, dass k_1 dem Lösungsschritt der Euler-Methode entspricht (2.31). Ein RK-Verfahren erster Ordnung wäre also äquivalent der Euler-Methode. Im Gegensatz dazu stellen die Einzelschritte bei höherer Ordnung k_n Teilschritte dar, über die anschließend gemittelt wird. Anschließend werden die Werte des aktuellen Zeitschrittes eingesetzt und somit der nächste Wert y_{i+1} angenähert. Für die Herleitung der Formeln zur Berechnung der Koeffizienten sei an dieser Stelle auf Sekundärliteratur verwiesen [23, 22].

3 Datensätze

Um zuerst die Anwendbarkeit der TE in Abschnitt 5.1 zu zeigen, verwendete ich zwei Testsysteme, die auch bereits von Hahs und Pethel [50], bzw. Boba et al. [15] vorgestellt wurden. Bei dem ersten System handelt es sich um eine sogenannte „Coupled Logistic Map“ (Abschnitt 3.1). Das zweite beruht auf einem „Hidden Markov Model“ (Abschnitt 3.2).

Anschließend folgte der Vergleich und die Bewertung der verschiedenen Maße anhand unterschiedlicher Beispieldatensätze, welche in drei Kategorien eingeteilt und in den Abschnitten 3.3 bis 3.5 beschrieben werden. Darauf wurden dann alle in Abschnitt 2.1 und 2.2 vorgestellten Maße zur Messung des Informationsflusses angewendet und verglichen.

Die erste Kategorie besteht aus einem einfachen Differenzialgleichungssystem, das zwei Oszillatoren koppelt. Die zweite Kategorie enthält zwei Subkategorien, deren Graphen unterschiedliche Topologien aufweisen: Barabási und Rényi [38, 8]. Auf diesen Topologien wurden dann Zeitreihen mithilfe eines Markov-Algorithmus erzeugt. Zum Schluss wurden verschieden komplexe Differentialgleichungssysteme aus der BioModels Database verwendet. Diese biologischen Netzwerke stellen zum Beispiel Stoffwechselprozesse oder Genexpressionskaskaden dar.

3.1 Gekoppelte logistische Gleichung (Coupled Logistic Map)

Wie von Hahs und Pethel [50] vorgestellt, wird die Transferentropie benutzt um eine Coupled Logistic Map zu analysieren. Diese Map basiert auf einer unidirektional gekoppelten chaotischen logistischen Gleichung (Logistic Map).

$$f(x) = r \cdot x \cdot (1 - x) \tag{3.1}$$

Der Parameter r wird dabei auf 4 festgelegt, wodurch die Logistic Map chaotisches Verhalten zeigt [5]. Das gekoppelte System kann dann wie folgt beschrieben werden:

$$\begin{aligned} x_{t+1} &= f(x_t) \\ y_{t+1} &= (1 - \xi) \cdot f(y_t) + \xi \cdot g_\alpha(x_t) \end{aligned} \tag{3.2}$$

Dabei stellt x_t die chaotische Logistic Map dar und wird als antreibendes System bezeichnet („Drive“). Das Empfängersystem y_t beinhaltet zwei Faktoren: 1.) Die Kopp-

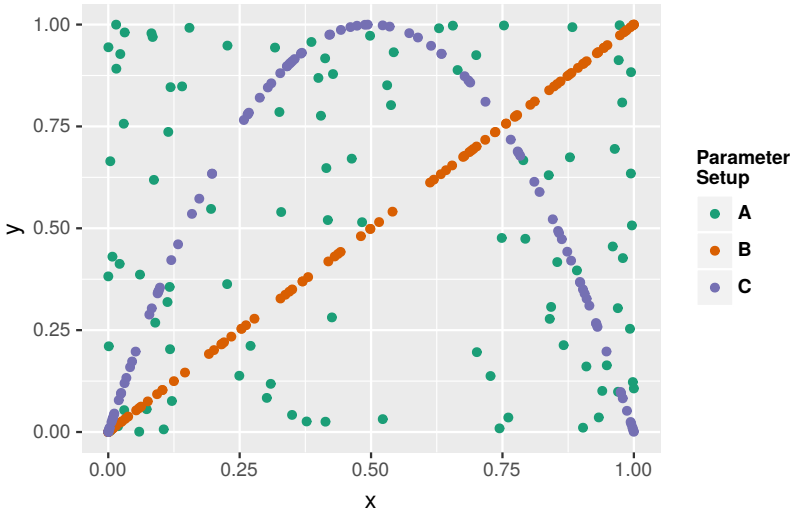


Abbildung 3.1: Die Abbildung zeigt die Korrelation zwischen x und y (Gleichung 3.2). Setup A (grün) zeigt völlig unabhängige Dynamik des Systems ($\alpha = 0$, $\xi = 0$). B (rot) zeigt $\alpha = 0$ und $\xi = 1$, wodurch das gekoppelte System x_x auf y_n direkt abgebildet wird, während es in C (lila) y_{n+1} vorhersagt ($\alpha = 1$, $\xi = 1$).

lungsstärke in Form von Parameter ξ (definiert im Bereich [0,1]). 2.) Die Kopplungsfunktion g_α modelliert das antizipatorische Element. Daraus ergibt sich Gleichung 3.3 wie folgt:

$$g_\alpha(x) = (1 - \alpha) \cdot f(x) + \alpha \cdot f(f(x)) \tag{3.3}$$

Parameter $\alpha \in [0, 1]$ der Funktion g_α moduliert nun die Vorhersagequalität in Bezug auf den Drive x. Werden ξ und α auf eins gesetzt, sagt x_t den Zustand von y_{t+1} exakt vorher. Mit einem Wert von $\xi = 0$ sind beide Systeme vollständig entkoppelt und unkorreliert. Abbildung 3.1 zeigt den Zusammenhang zwischen x, y und den Parametern ξ, α.

3.2 Hidden Markov Model

Im Gegensatz zum deterministischen System aus 3.1 handelt es sich beim Hidden Markov Model (HMM) um einen probabilistisches Modell. Bei einem Hidden Markov Model handelt es sich um eine Markov-Kette, die mit unbekannten Zuständen modelliert wird.

Eine Markov-Kette [49] ist ein stochastischer Prozess. Dabei wird der *aktuelle* Zustand x_t des Systems nur von dem *vorigen* Zustand x_{t-1} bestimmt. Dieser Prozess erster Ordnung wird auch als Markov-Eigenschaft bezeichnet. Man spricht in diesem Zusammenhang auch von einer gedächtnislosen Quelle. Ist der aktuelle Zustand ebenso von weiteren Zuständen der Vergangenheit abhängig, spricht man von einer Markov-Kette der Ordnung $n + 1$, entsprechend der Anzahl t berücksichtigter Vorgänger. Somit ergibt sich für den Übergang von Zustand s_i nach s_j die Wahrscheinlichkeit:

$$p_{i \to j}(t) = p(x_t = s_j | x_{t-1} = s_i), \qquad i, j = 1, ..., m \tag{3.4}$$

Dabei sind i und j hier diskrete abzählbare Zustände, die das System einnehmen kann.

Bei einem Hidden Markov Model wird nun angenommen, dass die beobachteten externen Zustände (auch als Emissionen bezeichnet) an eine verborgene (englisch „hidden“) Markov-Kette gekoppelt sind. Der interne Prozess ist dabei eine Markov-Prozess erster Ordnung und der aktuelle Zustand hängt von seinem vorherigen ab. Die Emissionen wiederum hängen nur von dem aktuellen Wert des internen Prozesses ab.

In dem hier beschriebenen Hidden Markov Model hat die interne Markov-Kette entweder zwei oder vier Zustände: $x_t \in \{\mathrm{A}, \mathrm{B}\}$ oder $\tilde{x}_t \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$. Anhand des internen Zustands werden in beiden Fällen zwei Zustände $y_t, \tilde{y}_t \in \{\ \mathrm{a},\ \mathrm{b}\}$ emittiert. In den Tabelle 3.1 und 3.2 befinden sich die Übergangs- und Emissionswahrscheinlichkeiten (ω, bzw. σ) für die beiden Modelle.

	x_t =A	x_t =B
x_{t-1} =A	0,9	0,1
x_{t-1} =B	0,1	0,9

(a) Übergang $\omega\,(x_t|x_{t-1})$

	y_t =a	y_t =b
x_t =A	0,9	0,1
x_t =B	0,1	0,9

(b) Emission $\sigma\,(x_t)$

Tabelle 3.1: Übergangs- $\omega\,(x_t|x_{t-1})$ und Emissionswahrscheinlichkeiten $\sigma\,(x_t)$ des ersten Hidden Markov Models.

Das zweite Modell verwendet doppelt so viele interne Zustände 3.3a, wie das Erste. Jedoch summieren sich die Wahrscheinlichkeiten für das zweite System derlei, dass sie den Emissionen des ersten entsprechen (vgl. Tabelle 3.2b und 3.3b) unter der Annahme, dass $t \to \infty$. Es ist anzumerken, dass dies aufgrund des Satzes von Perron-Frobenius und der Tatsache, dass die Matrizen der Übergangswahrscheinlichkeiten der beiden Systeme stochastisch sind, gültig ist.

Zwei Fragestellungen werden mit diesem Ansatz untersucht: Zum einen ob man die Transferentropie verwenden kann, um anhand der emittierten Zustände Informationen über den internen Zustand zu gewinnen und zum anderen, ob der interne Aufbau der Signalquelle einen Einfluss auf die Transferentropie hat und in wie weit die Signifikanzabschätzung mittels des Z-Scores davon abhängt.

3.3 Differentialgleichungssystem

Das in Abbildung 3.2 gezeigte Gleichungssystem koppelt zwei einfache Oszillatoren (u und v, sowie x und y). Die Variablen v und y werden jedoch ebenfalls gekoppelt, sodass ein Oszillator den anderen beeinflusst. Das Gleichungssystem ist in Formel 3.5 notiert. Dabei wurden verschiedene Stärken der Kopplungskonstanten k betrachtet.

In Abbildung 3.3 sind die Zeitreihen dieses Systems für vier verschiedene k über einen Zeitraum von 100 Zeitschritten abgebildet. Die Systeme wurden nach der Berechnung

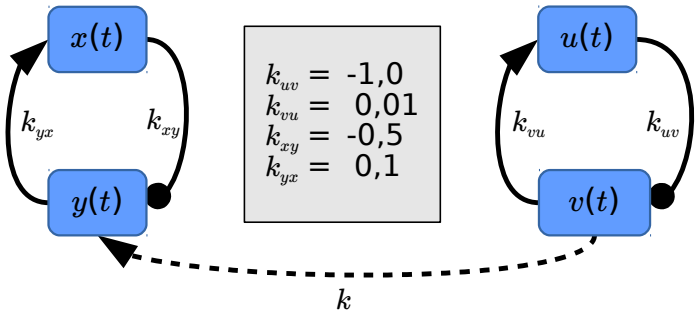
	$\tilde{x}_t$ =A	$\tilde{x}_t$ =B	$\tilde{x}_t$ =C	$\tilde{x}_t$ =D
$\tilde{x}_{t-1}$ =A	0,45	0,45	0,05	0,05
$\tilde{x}_{t-1}$ =B	0,45	0,45	0,05	0,05
$\tilde{x}_{t-1}$ =C	0,1	0,1	0,4	0,4
$\tilde{x}_{t-1}$ =D	0,1	0,1	0,4	0,4

(a) Übergang $\omega\left(\tilde{x}_t|\tilde{x}_{t-1}\right)$

	$\tilde{y}_t$ =a	$\tilde{y}_t$ =b
$\tilde{x}_t$ =A	0,9	0,1
$\tilde{x}_t$ =B	0,9	0,1
$\tilde{x}_t$ =C	0,1	0,9
$\tilde{x}_t$ =D	0,1	0,9

(b) Emission $\sigma\left(\tilde{x}_t\right)$

Tabelle 3.2: Übergangs- $\omega\left(\tilde{x}_t|\tilde{x}_{t-1}\right)$ und Emissionswahrscheinlichkeiten $\sigma\left(\tilde{x}_t\right)$ des zweiten Hidden Markov Models.



$$\begin{aligned}
\frac{\mathrm{d}u}{\mathrm{d}t} &= k_{vu}\, v\, \mathrm{d}t \\
\frac{\mathrm{d}v}{\mathrm{d}t} &= k_{uv}\, u\, \mathrm{d}t \\
\frac{\mathrm{d}x}{\mathrm{d}t} &= k_{yx}\, y\, \mathrm{d}t \\
\frac{\mathrm{d}y}{\mathrm{d}t} &= (k_{xy}\, x + k_{vy}\, v)\mathrm{d}t
\end{aligned} \tag{3.5}$$

Abbildung 3.2: Der Graph des einfachen DGL-Systems aus der daneben stehenden Formel 3.5. Die gewählten Werte für k (0, 0,001, 1, 10) drücken die Stärke der Kopplung der beiden Oszillatoren aus.

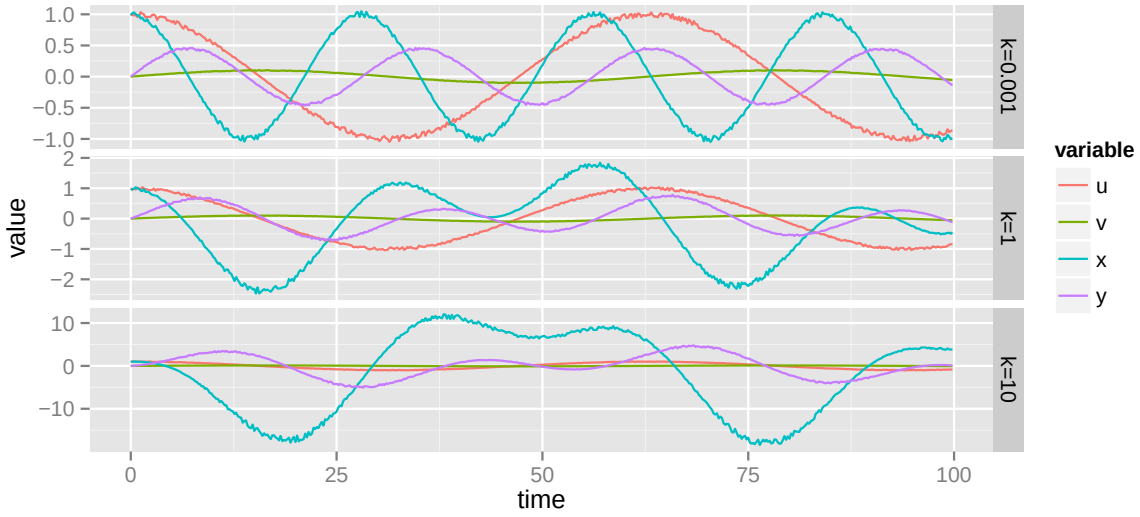


Abbildung 3.3: Simulation der Zeitreihen des Gleichungssystems 3.5. Die Simulationen wurden mit den im jeweiligen Panel dargestellten Kopplungskonstanten berechnet (Oben 0.001, Mitte 1.0, Unten 10.0). Anschließend wurden die Werte mit fünf Prozent additivem Rauschen versehen.

mit einem relativem Rauschen (Gleichverteilung) von 0,05 versehen, was Störungen bei realen Daten widerspiegeln soll, wie wechselnde Umweltbedingungen oder Messfehler. Anhand der Variable x lässt sich gut der Einfluss der Kopplungskonstante k erkennen.

Für die Berechnung der Transferentropie ist eine Diskretisierung (siehe Binning in Abschnitt 2.3) der Zeitreihen notwendig. Je mehr Bins man dafür wählt, umso höher ist die Auflösung der tatsächlichen Werte. Jedoch schlägt sich dies nicht notwendigerweise in einem besser feststellbaren Informationsfluss nieder. In vorangegangen Untersuchungen hatte sich eine Auswahl von vier bis 16 Bins als sinnvoll erwiesen, da diese zumeist einen guten Kompromiss zwischen erkennbarem Informationsfluss, Auflösung und Rechendauer darstellt.

3.4 Zeitreihen basiert auf Markov-Ketten

Ähnlich zu den in Abschnitt 3.2 vorgestellten Hidden Markov Models, verwendete ich als weitere Datengrundlage zum Vergleich der verschiedenen Maße, ein von mir entwickeltes Markov-Ketten-Netzwerk. An dieser Stelle sei erwähnt, dass im englischen Sprachraum bereits eine Definition von Markov-Netzwerken („Markov network“ oder auch „Markov random field“ [67]) besteht. In dieser Arbeit bezieht sich der Begriff „Markov-Netzwerk“ jedoch ausschließlich auf das von mir entworfene Modell. Für diesen Ansatz wurden zwei unterschiedliche Netzwerktopologien bei der Erzeugung der Graphen verwendet. In Abschnitt 3.4.1 werden kurz Grundlagen der Graphentheorie erläutert. Im darauf folgenden Abschnitt 3.4.2 befindet sich die Beschreibung der beiden von mir verwendeten Modelle. In Abschnitt 3.4.3 wird die Berechnung der einzelnen Knoten erklärt.

3.4.1 Graphentheorie

Die Graphentheorie beschäftigt sich mit den Eigenschaften und Beziehungen von Graphen. Viele Probleme lassen sich als Graphen darstellen. In praktisch allen wissenschaftlichen Bereichen (Geologie [92], Biologie [4, 19], Chemie [93] usw.) wird Graphentheorie daher verwendet um Zusammenhänge und Korrelationen zu fassen. Da die Rekonstruktion der Netzwerke letztliche bedeutet einen Graphen zu erzeugen, werden in diesem Abschnitt kurz die wichtigsten Begriffe im Zusammenhang mit der Graphentheorie erläutert [31].

Ein Graph besteht aus Knoten K (Vertices), die über Kanten E (Edges) verbunden sein können. Haben Kanten eine Richtung, spricht man von einem gerichteten Graphen; Sonst von einem ungerichteten Graphen (In dieser Arbeit werden nur gerichtete Graphen eingesetzt. Da der Informationsfluss entlang der Kanten analysiert wird ist die Richtung entscheidend). Wird Kanten eine Zahl zugewiesen spricht man von einem gewichteten Graphen. Graphen können ebenso als Adjazenz- bzw. Kontaktmatrix dargestellt werden. Dabei spiegeln die Spalten und Zeilen die Knoten wieder. In den Zellen der Matrix stehen dann binäre Werte im Falle eines ungewichteten Graphen oder die Kantengewichte.

Der Grad $d(k)$ eines Knotens k beschreibt die Anzahl der Kanten, die mit diesem Knoten verbunden sind. In einem gerichteten Graphen kann zwischen Eingangs- und Ausgangskanten unterschieden werden. Damit ergeben sich der Eingangs- $d^-(k)$ und Ausgangsgrad $d^+(k)$ für gerichtete Graphen.

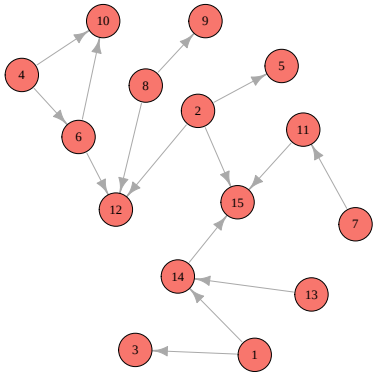
Es gibt weitere Elemente, die helfen einen Graphen zu beschreiben. Ein Pfad ist eine Abfolge von Kanten, die zwei Knoten verbindet. Um die Zentralität eines Graphen zu beschreiben, also wie stark die Knoten miteinander verbunden sind, hat sich auch im Deutschen der Begriff „Betweeness“ durchgesetzt [42]. Die Betweeness eines Knoten wird gemessen an der Anzahl der kürzesten Pfade, die durch diesen Knoten laufen. Als kürzester Pfad wird die geringstmögliche Anzahl an Kanten zwischen zwei Knoten bezeichnet. Ein weiteres Zentralitätsmaß ist die "Closeness". Sie beschreibt die aufsummierte Länge aller Kanten kürzester Pfade zwischen einem Knoten und allen anderen Knoten des Graphen [10]. Hat ein Pfad in einem Graphen den gleichen Knoten als Start und Ende spricht man von einem Zyklus, bzw. einem zyklischen Graphen.

3.4.2 Netzwerktopologien

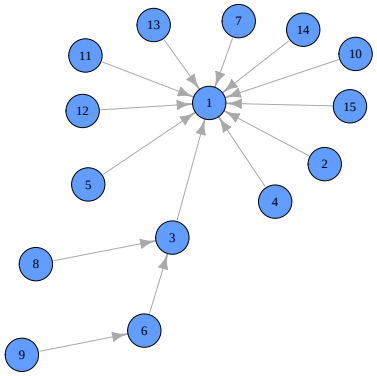
Die unterliegende Topologie der Markov-Netzwerke teilt sich in zwei Kategorien. In beiden Kategorien werden die Graphen zufällig erzeugt. Jedoch folgt die Verteilung der Kanten in der jeweiligen Kategorie einem anderen Schema, wodurch sich deren Topologien unterscheiden. Beispielgraphen für die beiden Ansätze sind in 3.4 zu sehen.

In der ersten Kategorie werden gerichtete Kanten nach dem Zufallsprinzip an eine festgelegte Anzahl Knoten angefügt (mit der Einschränkung, dass diese einen festgelegten maximalen Eingangsgrad haben). Für die Analyse werden Graphen mit 15 Knoten erzeugt. Der maximale Eingangsgrad ist entweder zwei oder drei. Die erzeugten Graphen ähneln dadurch einem Erdős-Rényi Modell [38] (vgl. 3.4a).

Es konnte gezeigt werden, dass biologische Netzwerke häufig einer skalenfreien Topolo-



(a) Zufälliger Graph ($d_{max}(k) = 3$) – Rényi-Modell



(b) Skalenfreier Graph – Barabási-Modell

Abbildung 3.4: Beispielgraphen der beiden Markov-Ketten-Modelle.

gie entsprechen [64]. Bei Netzwerken bezeichnet der Grad die Anzahl der Kanten die mit ihm verbunden sind. In diesem Fall folgt die Verteilung des Grads einem Potenzgesetz der Form:

$$P(d(k)) \propto d(k)^{-\gamma} \tag{3.6}$$

dabei modelliert der Parameter γ die Verteilung des Grads κ. Der Aufbau des Netzwerks wird dadurch auffällig von wenigen Knoten mit hohem Vernetzungsgrad bestimmt, während die meisten anderen Knoten nur einen geringen Grad aufweisen (siehe 3.4b, z. B. Knoten 1). Um die Graphen zu erzeugen, wurde die Methode von Barabási-Albert [8] aus dem igraph package [27] in `R` verwendet. Im Folgenden werden die beiden beschriebenen Netzwerktypen vereinfacht Rényi- und Barabási-Netzwerke genannt.

3.4.3 Berechnung der Zustände

Bei den verwendeten Ketten handelt es sich um homogene unendliche Ketten mit diskreten Zuständen und stationären Übergangswahrscheinlichkeiten. Die Anzahl der möglichen Zustände x, die ein Knoten k zum Zeitpunkt t einnehmen kann, wurde entweder auf vier oder acht festgelegt. Zur Erzeugung des Zustands für den Zeitpunkt $t+1$ wird zunächst festgestellt, ob und in welchen Zustand der Knoten wechselt. Die Wahrscheinlichkeiten für die jeweiligen Zustände ist gleich. Die Gesamtzahl N der Knoten wurde auf 15 pro erzeugtem System festgelegt. Für ein System mit n_Z verschiedenen Zuständen ergibt sich bei einer Übergangswahrscheinlichkeit p, die Wahrscheinlichkeit das ein Knoten sein Zustand x_t in einen bestimmten Zustand $\neq x_t$ ändert mit

$$p(x_{t+1} \neq x_t) \quad = \quad \frac{p}{n_Z - 1}. \tag{3.7}$$

Die Wahrscheinlichkeit $\tilde{p}$, dass der Zustand des Knotens unverändert bleibt, ist somit

$$\tilde{p} = p(x_{t+1} = x_t) = 1 - p. \tag{3.8}$$

Die Dynamik der Modelle resultiert aus zwei Eigenschaften und ist für beide Arten von Netzwerken identisch: Zum einen der Wahrscheinlichkeit p, dass sich der Zustand x eines Knotens k_m aufgrund seiner Eigendynamik ändert und zum anderen, dass der Zustand des Knotens aufgrund der Flip-Wahrscheinlichkeit ν durch einen der mit ihm verbundenen Knoten geändert wird. Sei $x_{m,t+1}$ der Zustand x des Knotens k_m zum Zeitpunkt $t+1$ und $x_{l,t}$ der Zustand des Knotens k_l zum Zeitpunkt t. Die Flip-Wahrscheinlichkeit für den Knoten k_m durch den Knoten k_l ist damit definiert als $\nu(x_{m,t+1} = x_{l,t})$. Für eine besserer Lesbarkeit wird fortan die vereinfachte Notation $\nu_{l \to m}$ für die Flip-Wahrscheinlichkeit und p_m für die Übergangswahrscheinlichkeit des Knotens m verwendet.

Anhand dieses zweistufigen Verfahrens, werden die Zeitreihen der einzelnen Knoten erzeugt. Der Entscheidungsbaum in Abbildung 3.5 verdeutlicht den Ablauf zur Erzeugung eines Zeitschritts für einen einzelnen Knoten.

Die Übergangswahrscheinlichkeiten für die Knoten wurden für Knoten ohne Ausgangskanten zufällig zwischen 0 und 0,4 gewählt. Für Knoten mit mindestens einer Ausgangskante wurden die Wahrscheinlichkeiten im Bereich zwischen 0,3 und 0,9 erzeugt. Dies soll gewährleisten, dass bei wenigen möglichen Zuständen der Knoten das Übertragen des Zustands von einem zum anderen Knoten messbar bleibt.

Die Flip-Wahrscheinlichkeit wurde für die Generierung der Zeitreihen auf 0,6, 0,7 oder 0,8 gesetzt, um einen mittleren bis starken Fremdeinfluss der Knoten zu simulieren. Hat ein Knoten mehrere eingehende Kanten, wird die Wahrscheinlichkeit des Flips gleichmäßig aufgeteilt. Der Aufbau dieses Systems soll dafür sorgen, dass die zeitliche Dynamik des Netzwerks stark von den Flip-Vorgängen bestimmt ist und somit den Informationsfluss im Netzwerk reflektieren.

Abbildung 3.6 veranschaulicht das zugrunde gelegte Prinzip. Dabei ist zu beachten, dass bei Eintreten eines Flips der Zustand des Knotens überschrieben wird, selbst wenn dieser bereits durch die eigene Übergangswahrscheinlichkeit seinen Zustand zuvor verändert hatte. Der geflippte Knoten nimmt damit im nächsten Zeitschritt den aktuellen Zustand des fremden Knotens an.

3.5 Biolgische Netzwerke - BioModels Database

Als Quelle für biologische Daten wurde die von EMBL-EBI angebotene BioModels Database ([75], www.ebi.ac.uk/biomodels-main) verwendet. Diese bietet, potentiell, eine große Auswahl an biologischen Netzen, die zum Teil auch von den Betreibern handkuriert sind. Die Netzwerke werden mit IDs der Form BIOMD0000000xxx angegeben. Um

	BM12	BM95	BM99	BM106	BM160
Iterationen	300000	100000	150000	100000	120000
Schrittgröße	0.005	0.0002	0.00025	0.0002	0.001
Stille Iterationen	200000	0	50000	0	20000

Tabelle 3.3: Übersicht der verwendeten Parameter für die Simulation der biologischen Netzwerke.

die Angaben zu vereinfachen, verwende ich eine verkürzte Notation, die nur maximal die letzten drei Stellen verwendet. Zum Beispiel wird die ID BIOMD0000000042 zu BM42 verkürzt.

Beim Konvertieren der Daten waren jedoch Inkonsistenzen der im SBML Format hinterlegten Daten feststellbar. Etwa weil die verwendete Nomenklatur für Parameter und Variablen nicht eingehalten wurde. Dadurch konnte es vorkommen, das Entitäten beim konvertieren in C-Code nicht korrekt übersetzt wurden. Dies äußerte sich in fehlenden Gleichungen oder Duplikaten. Da das manuelle Kurieren aller Dateien zu umfangreich gewesen wäre, wurden anhand verschiedener Filterschritte nur Netzwerke verwendet, die sich fehlerfrei konvertieren ließen. Anschließend wurden die Dateien noch einmal manuell geprüft (und evtl. doppelte Gleichungen entfernt).

Für die Simulation der Netzwerke wurde das in 4.1 beschriebene Tool *SOMNIBIEN* verwendet. Um einen realistischen Ablauf der Zeitreihen zu erzeugen, wurden die Parameter des Solvers entsprechend der Tabelle 3.3 eingestellt. Die betreffenden Parameter sind die Anzahl der Iterationen und Schrittgröße, sowie in manchen Fällen eine gewisse Anzahl an stillen Iterationen bis sich die Simulation stabilisiert hat (sog. Relaxatationszeit oder "Burn-In").

Die Simulationen der Netzwerke wurden zuerst als einfache Differenzialgleichungen (ODE) durchgeführt. Zusätzlich wurden die Netzwerke aber auch mit geringem und starken Rauschen der Daten simuliert, indem die Lösung einer stochastische Differenzialgleichung (SDE) durchgeführt wurde. Die Abbildung 3.7 zeigt beispielhaft Zeitreihen mit unterschiedlich starkem Rauschen. Dieser Ansatz unterscheidet sich daher von dem in Abschnitt 3.3 verwendeten additiven Rauschen.

Wie in Tabelle 3.3 zu sehen ist, wurden die Netzwerke mit einer ausreichend großen Anzahl an Schritten erzeugt. Von diesen umfangreichen Zeitreihen wurden anschließend kleinere Stichproben für die Berechnung der Informationsmaße entnommen.

Um die Fähigkeit zur Rekonstruktion der Netzwerktopologie der einzelnen Informationsmaße zu bestimmen, musste aus den vorhanden Daten zu den Netzwerken der BioModels Database eine Adjazenzmatrix erstellt werden. Dazu wurden die Differentialgleichungen und der Graph des Netzwerks herangezogen. Alle Metaboliten, die einen anderen Metaboliten direkt beeinflussen wurden als Kante in die Adjazenzmatrix übernommen. Allerdings wurde hierbei eine Vereinfachung vorgenommen. In dem Ausschnitt des Graphen zu BM12 (Abb. 3.8a) ist erkennbar, dass die Reaktionen, nicht direkt bei der Rekonstruktion berücksichtigt werden. Beispielsweise führt die Translation von cl mRNA direkt zu cl protein. Daraus ergibt sich für den genannten Subgraph die Kontaktmatrix 3.8b. Die Abbaureaktionen (orange) werden hingegen gar nicht berücksichtigt,

da sie an keinem Informationsfluss zwischen den Entitäten beteiligt sind.

Die Auswahl der Systeme erfolgte anhand deren Größe. Ziel war es verschiedene Größen in der Auswertung zu berücksichtigen. Das kleinste System (BM12) enthält sechs Knoten und, nach der Vereinfachung, sechs Kanten. Das größte Netzwerk besteht aus 19 Knoten und 64 Kanten. In den nächsten Teilabschnitten werden die verwendeten Netzwerke kurz beschrieben. Die Graphen der Netzwerke befinden sich entweder in den jeweiligen Abschnitten oder im Anhang.

Repressilator - Ein synthetisches oszillatorisches Netzwerk von Transkriptionsfaktoren [BM12]

Diese Netzwerk (Abb. 3.9) stellt ein synthetisches Netzwerk in *Escherichia coli* dar. Die Autoren [36] beschreiben ein transfiziertes System aus drei Transkriptionsrepressoren, welches an ein GFP(Green Fluorescent Protein) Reportersystem gekoppelt ist und periodisch GFP produziert. Somit kann das zyklische Verhalten des, wie die Autoren es nennen, Repressilator zeitlich erfasst werden. Die Dauer eines Zyklus überschreitet dabei die Länge des Zellzyklus des Bakteriums. Die Auswertung des Fluoreszenzsignals lies somit gleichzeitig die stabile Transfektion des Bakteriums beobachten, da das Signal über mehrere Generationen hinweg zu beobachten war.

Modellierung des zirkadianen Rhythmus in *Arabidopsis* [BM95]

Zeilinger et al. [125] beschreiben in ihrer Publikation den zirkadianen Rhythmus von *Arabidopsis*, den sie im Vergleich zu früheren Veröffentlichungen um die Pseudo-Response Regulatoren PRR7 und PRR9 erweitern. Das Modell (Abbildung 8.1 im Anhang) wurde anhand experimenteller Daten entworfen und anschließend mittels eines Algorithmus optimiert. Einer Validierung des Netzwerks fand anhand mehrerer phänotypischer Mutanten statt.

Spontan oszillierende Zellen in *Dictyostelium* [BM99]

Das von Laub et al. [73] beschriebene Netzwerk (Abbildung 8.2 im Anhang) simuliert die Interaktion verschiedener Proteine in *Dictyostelium* Zellen. Die Oszillation tritt in homogenen Populationen nach vier Stunden auf. Der Ablauf startet mit extrazellulärem cAMP, welches an CAR1 bindet und somit ERK2 und dieses wiederum ACA aktiviert. ACA aktiviert PKA (und somit weitere Genexpression), die in einer Rückkopplung nun CAR1 sowie ERK2 inhibiert. ERK2 inhibiert gleichzeitig auch REG A (Abbau von cAMP) und damit indirekt auch PKA.

Simulation des Arachidonsäurepfads [BM106]

In der Arbeit von [123] beschäftigen sich die Autoren mit einem metabolischen Netzwerk, das in die Mediation von Entzündungsprozessen involviert ist, der sogenannte Arachidonsäurepfad. Ziel der Studie war es, ein besseres Verständnis der Dynamik des Netzwerks

zu erlangen, um dadurch mögliche Ansätze für die Medikamentenentwicklung zu finden. Der Graph für das Netzwerk ist im Anhang in Abbildung 8.3 gezeigt.

Modellierung des zirkadianen Rhythmus in *Drosophila* [BM160]

Dieses Modell in [122] stellt das Netzwerk der Transkriptionsregulation für die damaligen bekannten Faktoren des zirkadianen Rhythmus in *Drosophila melanogaster* dar. Anstatt Hill Funktionen für die Genregulation anzunehmen, modelliert das Modell (Abbildung 8.4 im Anhang) explizit die Interaktion von Transkriptionsfaktoren und Promotoren. Es ist zwar in der Lage sowohl vollständige Dunkelzyklen als auch hell/dunkel Wechsel robust zu simulieren, allerdings gibt es Abweichungen zu Experimentaldaten.

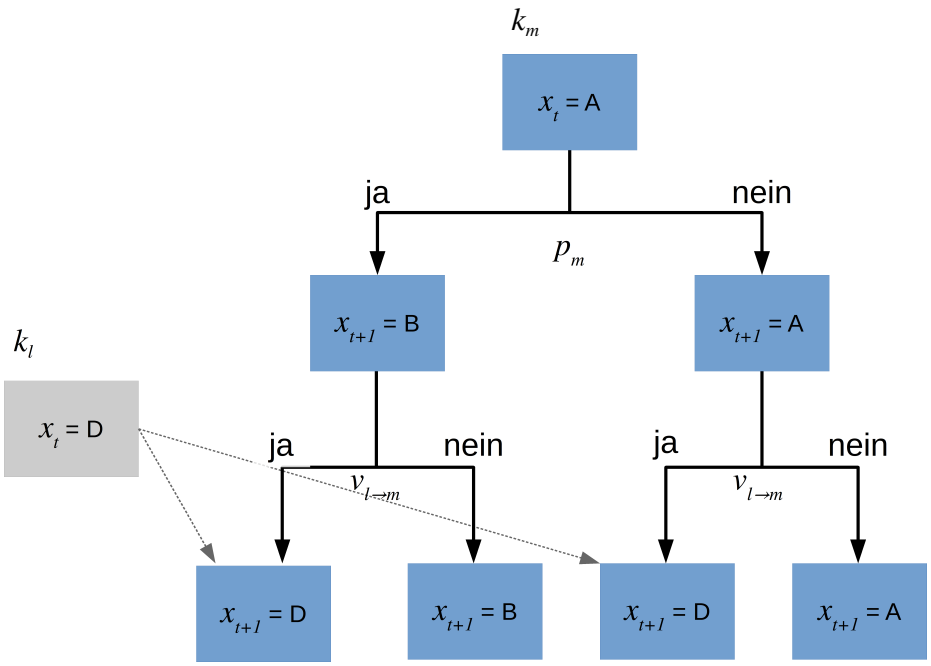
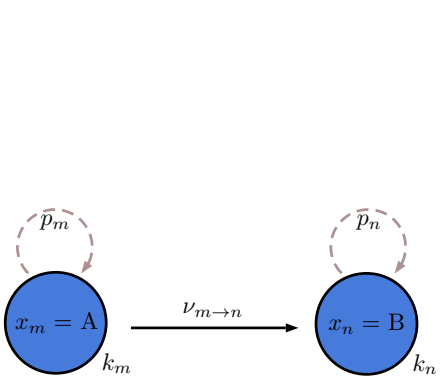
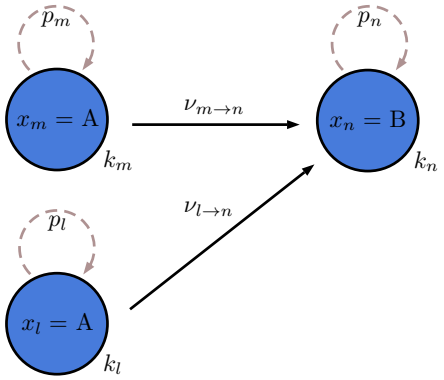


Abbildung 3.5: Beispiel eines Entscheidungsbaum für die Zustandsänderungen der Knoten mit Eingangskanten in den konstruierten Markov-Netzen. Der Entscheidungsbaum verdeutlicht nur das zweistufige Verfahren um den Zustand für den nächsten Zeitpunkt in der Zeitreihe des Knoten k_m zu bestimmen (blaue). Angenommen sei ein System das vier verschiedene Zustände ($x \in \{A; B; C; D\}$) annehmen kann. Zunächst werden die Übergänge des Knoten k_m mit der Wahrscheinlichkeit p_m festgestellt. Die Notation „Ja“ an den Pfeilen deutet an, dass der Knoten seinen Zustand geändert hat. Bei einem „Nein“ verbleibt der Knoten in dem aktuellen Zustand. Anschließend kann der Knoten nochmals aufgrund der Flip-Wahrscheinlichkeit $\nu_{l\to m}$ durch den Zustand des Knoten k_l überschrieben werden (grau).

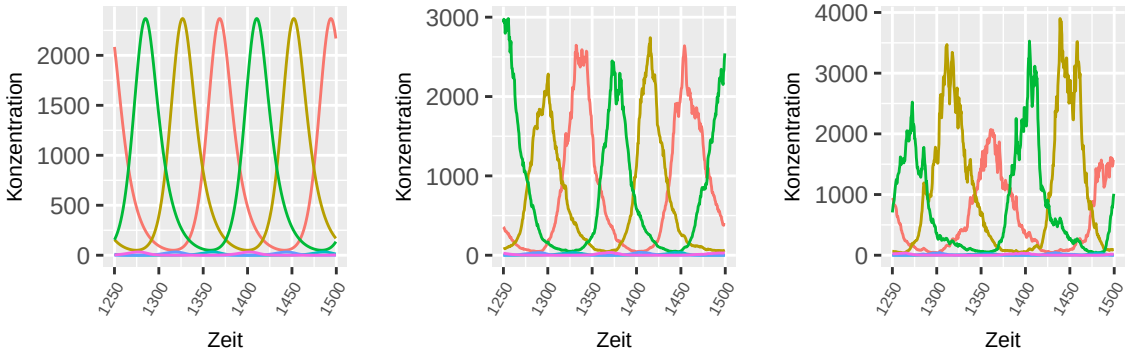


(a) Prinzip des Markov-Netzwerks für den Flip durch einen fremden Knoten.



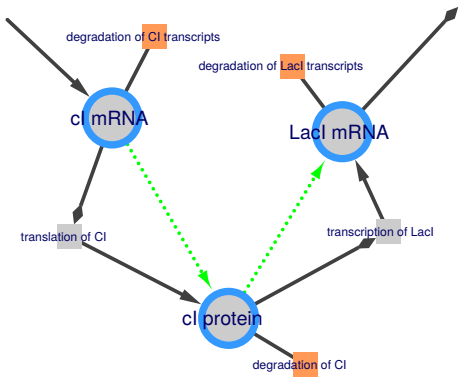
(b) Prinzip des Markov-Netzwerks für den Flip durch zwei fremde Knoten.

Abbildung 3.6: Prinzip des Markov-Netzwerks. (a) Über die Übergangswahrscheinlichkeit p bestimmt ein Knoten seinen nächsten eigenen Zustand x selbst (grau gestrichelter Pfeil). Des Weiteren können Knoten auch den Zustand anderer Knoten über ihre Flip-Wahrscheinlichkeit ν ändern (durchgehender schwarzer Pfeil). (b) Außerdem kann der Knoten k_m seinen Zustand x_m mit einer Wahrscheinlichkeit von $\nu_{m\to n}(x_j)$ auf den Knoten k_n übertragen. Wird ein Knoten von mehreren anderen Knoten beeinflusst, ist die Flip-Wahrscheinlichkeit gleichverteilt über alle Eingangskanten. Die Wahrscheinlichkeiten p der verschiedenen Datensätze sind in Tabelle 3.1 zu finden (Datensatz 1).



(a) Kein Rauschen (ODE) (b) Geringes Rauschen (SDE) (c) Starkes Rauschen (SDE)

Abbildung 3.7: Zeitreihen der Simulationen von BM12 mit unterschiedlichem Rauschterm.



(a) Ausschnitt des Netzwerkgraphen von BM12.

	cI protein	LacI mRNA	cI mRNA
cI protein	0	0	1
LacI mRNA	1	0	0
cI mRNA	0	0	0

(b) Adjazenzmatrix des Auschnitts von BM12.

Abbildung 3.8: Die grünen gepunkteten Pfeile im Graphen (a) stellen eine Kante in der Adjazenzmatrix (b) dar. Die orange gekennzeichneten Abbaureaktionen werden bei der Erstellung der Matrix nicht berücksichtigt.

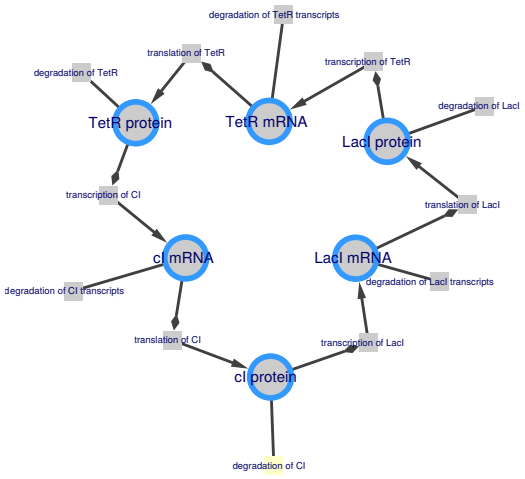


Abbildung 3.9: Vollständiger Graph des Netzwerks BM12

4 Software und Tools

4.1 R-Paket: SOMNIBIEN

Für das Ziel dieser Arbeit benötigten wir zunächst eine Möglichkeit um die Netzwerke aus der BioModels Database zu simulieren. Folgende Anforderungen waren dabei wichtig:

- Simulation stochastischen Differentialgleichungen.
- Import der Daten aus der BioModels Database.
- Integration mit R

Für einige dieser Punkte gibt es bereits Tools (auch in R), die einen Teil dieser Aufgaben übernehmen. Theoretisch wäre es also möglich den gesamten Prozess auf diese einzelnen Programme zu verteilen. Jedoch gelang es nie die individuellen Elemente (u.a. deSolve [106], SBMLR [96], rsbml [74]) so nahtlos zu verzahnen, dass die komplette Pipeline für die Durchführung verfügbar war. Die Interaktion zwischen den verschiedenen getesteten Bestandteilen sorgte für Abstürze des Solvers oder die unterschiedlichen Formatrevisionen von SBML [39, 40] stellten sich als inkompatibel, bzw. dessen Inhalt fehlerhaft festgehalten dar.

Daher entschieden wir uns für den Solver von Daniel Steck [110] als numerischen Integrator, da dieser sich im Rahmen der Entwicklung dieses Pakets als verlässlich erwiesen hatte [48]. Der Solver wird zusammen mit in C geschriebenen Differentialgleichungen kompiliert und ausgeführt.

Daher benötigten wir für den zweiten Punkt nun ein Tool, welches die Differentialgleichungen der Netzwerke von ihrem nativen XML Format in eine C-Code Datei konvertiert. Die Anwendung COPASI [61] bot diese Möglichkeit. Da es sich im Batch-Modus ansteuern lässt, konnte es problemlos in das R-Pakets integriert werden. COPASI (Complex Pathway Simulator) ist ein Tool, welches, neben vielen weiteren Funktionen wie z. B. Flux-Balance Analysis (FBA), die Zeitreihen Simulation von biologischen Netzwerken erlaubt [61]. Dieser Teil von COPASI funktionierte jedoch nicht, wie erwartet mit den von uns angestrebten stochastischen Differentialgleichungen, sodass weiterhin der bereits oben erwähnte Solver zum Einsatz kam.

Neben COPASI gibt es noch weitere Programme [25], die die Simulation von biologischen Netzwerken ermöglichen. CellDesigner, zum Beispiel, [43] ist nicht nur ein Simulationstool. Es ist vor allem darauf ausgelegt biologische Netzwerke darzustellen und zu modellieren. Das native Dateiformat für CellDesigner ist die Systems Biology Markup Language (SBML) [39]. Es handelt sich dabei um ein XML Format zur Beschreibung biologischer Systeme. Cytoscape [104] ist ebenfalls ein verbreitetes Tool zum

Bearbeiten und Erstellen von biomolekularen Netzwerken. Diese Open-Source-Software zeichnet sich dadurch aus, dass sich durch die Plug-in-Fähigkeit die Funktionalität mittels Erweiterungen vergrößern lässt [101]. Neben den UI-getriebenen Ansätzen bietet z.B. PySB [79] ein Framework für die Programmiersprache Python um biochemische Systeme zu simulieren. Eine weitere Herausforderung beim Verarbeiten der Netzwerke stellen die verschiedenen Versionen von SBML [40] dar: Da nicht alle Tools sofort das höchste SBML-Level unterstützen, kann es hier zu Problemen beim Einlesen der Daten kommen.

Da sich Wet-Lab-Biologen und Biotechnologen üblicherweise interaktiv mit ihren Systemen auseinandersetzen, war es unser Ziel das Paket möglichst einfach in der Benutzung zu gestalten. Gleichzeitig ist der beteiligte Parameterraum hochdimensional und kann hohe Rechenleistung erfordern, die nur von HPC- Infrastrukturen geboten wird. Deshalb entschlossen wir uns, beide Ansätze in ein System zu integrieren, um dem Nutzer sowohl die Möglichkeit des Batchbetriebs oder der interaktiven Analyse in R zu bieten.

Um nun die Simulation der verwendeten biologischen Netzwerke mit stochastischen Differentialgleichungen zu realisieren, wurde von mir die Entwicklung eines `R`-Pakets durch einen Studenten betreut [48]. Das Paket trägt den Namen SOMNIBIEN (Simulation Of Metabolic Networks Influenced By Internal And External Noise) und ermöglicht eine effiziente Simulation mittels eines SDE Solvers in Fortran Code. Zusätzlich wurde es durch die verwendete Datenbank (BioModels Database) notwendig COPASI als Konverter zu benutzen, um die so erzeugten C-Dateien zu importieren. Das Listing 4.1 zeigt einen typischen Workflow, wie er in SOMNIBIEN durch geführt werden kann.

Die Ergebnisse und Erkenntnisse aus der Entwicklung dieses `R`-Pakets wurden als Beitrag zur Informatik 2014 eingereicht [16] und dort in einem Vortrag vorgestellt. Die folgende Abschnitt nimmt Bezug auf diese Publikation.

Der Aufbau des `R`-Pakets kann Abbildung 4.1 entnommen werden. Das Kompilieren muss für jedes Netzwerk neu erfolgen. Allerdings kann die Bibliothek mit verschiedenen Parameter gestartet werden, die Einfluss auf die Simulation nehmen.

Aufbau des Pakets

Der Idee des Pakets lag der Anspruch zugrunde eine Lösung für `R` zu entwickeln, welche die Geschwindigkeitsvorteile von Fortran nutzt ohne die Flexibilität von `R` zu opfern. Zu diesem Zweck wurde der Aufbau wie in Abbildung 4.1 entworfen. SOMNIBIEN kapselt dazu die einzelnen Anwendungsteile verschiedener Sprachen und ermöglicht somit dem Benutzer ohne `R` zu velassen:

- Automatische Konvertierung des Dateiformats.
- (Stochastische) Simulation der Netzwerke. SOMNIBIEN kompiliert im Hintergrund die einzelnen Code Bestandteile und macht sie in `R` verfügbar.
- Direkte Analyse der Daten mit verfügbaren Tools in `R`.

Durch die Möglichkeit direkt auf die BioModels Database zu zugreifen, kann der Nutzer direkt Modelle in seinen Workspace laden. Um die Netzwerke zu simulieren muss

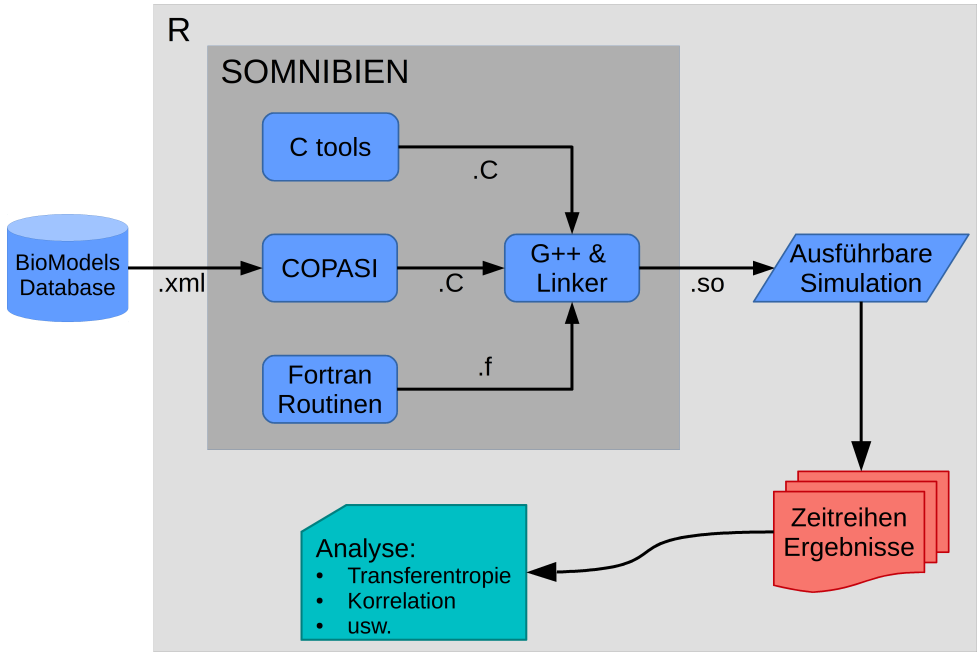


Abbildung 4.1: Aufbau des SOMNIBIEN Pakets. Das Netzwerk im XML Format wird von COPASI in C-Code umgewandelt. Der `G++` Compiler erzeugt aus dem Fortran Solver zusammen mit weiteren Code Elementen ein *shared object*. Dieses wiederum kann in `R` verwendet werden, um damit Zeitreihen zu erzeugen.

dem Solver die Datei allerdings in einem speziellen C-Code vorliegen. Dafür wird auf den Kommandozeilenaufruf von COPASI zurück gegriffen und die Datei im Hintergrund konvertiert. Der Vorteil ist, dass der Benutzer weiterhin seinen vollständigen Arbeitsablauf in `R` durchführen kann. Der Code in Listing 4.1 zeigt den Ablauf einer Simulation mit SOMNIBIEN. Anstatt mehrere verschiedene Tools zu verwenden, kann die Simulation mit wenigen Befehlen gestartet werden.

Problematik des `R`-Foreign Interface

Bei der Implementierung des Solver-Interface ergaben sich Schwierigkeiten, die darauf zurück zu führen sind, wie `R` auf externe Programmbibliotheken (Shared Objects) zugreift. Es folgten mehrere Iterationen, bis die finale Strukturierung des Pakets umgesetzt werden konnte (Details in [16]).

`R` behandelt intern Funktionssymbole von Bibliotheken wiederum als Objekt. Dadurch können keine Zeiger auf C- oder Fortran-Onjekte in `R` aufgelöst werden und Funktionszeiger an den SDERK-Solver zu übergeben ist somit unmöglich. Als weiteres Problem stellt sich das Fortran Interface zu `R` dar, welches nur primitive Datentypen übergeben kann. Dadurch ist es nicht möglich Funktionen oder Pointer auf selbige zu übergeben.

Dies machte die Entwicklung weiteren Codes (im Folgenden C-Controller genannt) notwendig. Dieser dient der „Vermittlung“ zwischen den eingesetzten Programmierspra-

Auflistung 4.1: Typischer Ablauf einer Analyse mit SOMNIBIEN

```
library("SOMNIBIEN")

## Es wird die SBML Datei in eine C-Code konvertiert und ein "shared object" erzeugt.
PrepareODEFile("BIOMD0000000043.xml", filename="Bio043")

## Aus dem C-Code werden die für die Simulation nötigen Konzentrationen und Parameter
     extrahiert.
InitConc <- GetCopasiInitialValues("BIOMD0000000043.c")
Parameters <- GetCopasiParameterValues("BIOMD0000000043.c")

## Starten der Simulation erzeugt eine Tabelle, die jeweiligen Konzentrationen als
    Zeitreihe in den Spalten enthält.
SimData <- StartSim(sfile="Bio043.so", parameters=Parameters,
concentrations=InitConc, stepwidth=0.01)

## Die Zeitreihen können unterbrechungsfrei mit R weiter untersucht werden.
```

chen und umgeht die oben genannten Schwachstellen des R-Interface. Controller und Solver werden von dem G++ Compiler [109] aus der GNU Compiler Collection (GCC) zusammen in eine gemeinsame Programmbibliothek kompiliert. Dies geschieht während der Installation des R-Pakets.

Wenn der Benutzer anschließend eine Netzwerkmodell in R öffnet, wird eine Bibliothek generiert. Diese SDE-Bibliothek wird von dem C-Controller dynamisch geladen, der wiederum die Funktionspointer an den Solver übergibt. Abbildung 4.2 veranschaulicht die Struktur des Pakets.

Im Anhang befindet sich ein Code-Ausschnitt (Auflistung 8.2), der veranschaulicht wie die Pointer-Arithmetik und die Konvertierung für das Interface vonstatten geht. Durch das Verschmelzen der einzelnen oben genannten Bestandteile zu einem R-Paket kann der vollständige Workflow nun ebenfalls von Wissenschaftlern ohne Expertenwissen im Bereich der Programmierung bedient werden. Außerdem bietet die Integration in R flexible Analysemöglichkeiten.

Parallelisierung

Um die Leistungsfähigkeit der Software weiter zu verbessern, wurde von uns an geeigneten Stellen des Arbeitsablaufs Parallelisierungen des Codes vorgenommen.

Zum einen wurde das Rauschen, welches in Formel 2.29 durch die Parameter $v_i^{\downarrow}$ und $v_i^{\uparrow}$ ausgedrückt wird, parallelisiert. Die einzelnen Simulationen sind unabhängig voneinander (SIMD - Single Instruction, Multiple Data) und die Parallelisierung kann direkt in R mit vorhanden Paketen (u.a. `doParallel`) realisiert werden.

Es gibt ein weiteres Szenario das die Prozessoptimierung durch Parallelisierung verspricht. Bei der Optimierung und dem Design von Netzwerken beschäftigt man sich häufig mit nur einem System und damit einem Parametersatz gleichzeitig (z. B. Simu-

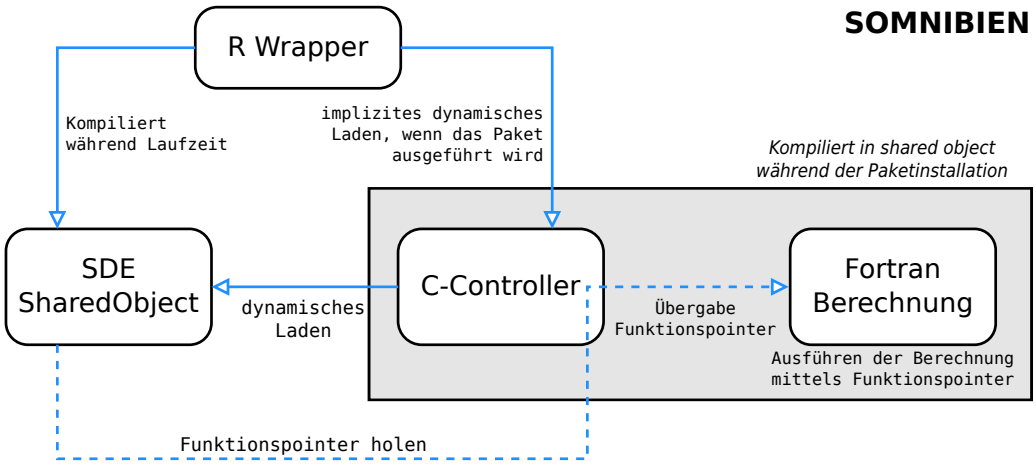


Abbildung 4.2: Schema der finalen Implementierung für SOMNIBIEN. Der Fortran Solver und der C-Controller werden in eine gemeinsame Bibliothek kompiliert (graue Box). Diese Programmbibliothek wird beim Starten des `R`-Pakets geladen. Für jedes zu simulierende Netzwerk wird anschließend während der Laufzeit eine eigene Bibliothek (SDE Shared Object) erzeugt, welche dynamisch geladen wird. Beim Starten einer Simulation übergibt der C-Controller von der SDE Bibliothek einen Pointer zur SDE Funktion an den Fortran Code des Solvers.

lated Annealing [69]). Das Optimierungsverfahren müsste dann nur eine Variante eines Systems berechnen, eine neue mit Veränderung der Parameter (oder Graph Topologie) erstellen, anschließend die Eigenschaften über Simulation bestimmen und so weiter. Hierbei wäre eine High-Level-Parallelisierung wie vorher nicht effizient, da die Daten nicht parallel bearbeitet werden können. Deswegen haben wir ebenfalls versucht mittels dem OpenMP [30] Framework eine Parallelisierung der rechten Seite der DGL zu erreichen.

In vielen Fällen wurde bereits eine mögliche Parallelisierbarkeit für Solver von Differenzialgleichungen vorgeschlagen [1, 87]. Allerdings war eine solche Optimierung des Runge-Kutta-Algorithmus ohne weiteres nicht direkt möglich. Die Vorhersage- und Korrekturschritte öffnen und schließen (zumindest anhand der hier verwendeten Daten) in kurzer Zeit zu viele Threads, sodass dadurch in einigen Fällen ein sehr hoher Overhead entsteht. Die Abbildung 4.3 zeigt die dazugehörige Auswertung. Da diese Parallelisierung die Rechenzeit oft nur wenig herab setzte (manchmal sogar erhöhte), wurde der Ansatz schlussendlich wieder verworfen.

4.2 `C++` Bibliothek zur Berechnung der Transferentropie

Ein weiteres von mir betreutes Studentenpraktikum diente der Implementierung einer `C++` Bibliothek um die Transferentropie möglichst schnell zu berechnen (libtransentr).

ID	Metabolite	Parameter	Reaktionen
BM042	15	25	25
BM160	19	47	41
BM209	6	49	12
BM422	22	13	43
BM505	45	140	59

Tabelle 4.1: Übersicht der getesteten Netzwerke. Für den Vergleich wurden verschiedene Netzwerke mit unterschiedlicher Topologie und Anzahl der Parameter herangezogen.

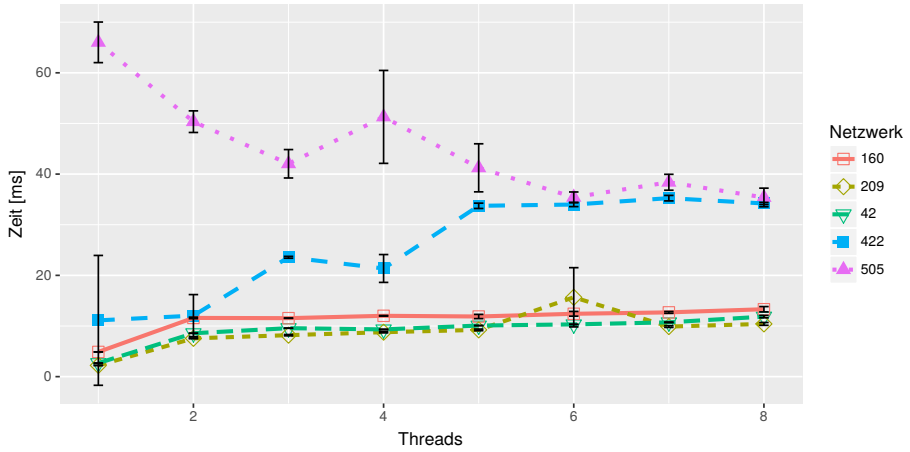


Abbildung 4.3: Mittlere Rechenzeit eines Integrationsschritts für fünf verschiedene Netzwerke aus Tabelle 4.1. Das Parameterrauschen β beträgt 0.01 (vgl. 2.29). Fehlerbalken sind die Standardabweichung aus 100 Wiederholungen der Simulationen.

Der Fokus war hierbei eine effiziente Datenstruktur zu verwenden. Das Paket ist weiterhin in der Lage ein Z-Score (siehe Abschnitt 2.4.2) basiertes Nullmodell zu berechnen. Letzteres wurde zusätzlich mit OpenMP parallelisiert um die Rechenzeit zu verkürzen. Die Bibliothek wurde von bereits von Boba et al. [15] publiziert.

Wie bereits in Abschnitt 2.2.5 erwähnt, lässt sich mit Hilfe der Transferentropie die Richtung des Informationsflusses feststellen. Es fehlt jedoch ein statistischer Maßstab für dieses Ergebnis. Ich entschied mich daher, die Methode der sog. Z-Scores zur Einschätzung der statistischen Signifikanz zu verwenden. Diese wurde erfolgreich in verschiedenen Arbeiten als Signifikanzmaß für die Transinformation (MI) eingesetzt [118, 52]. Eine Definition des Z-Score befindet sich in 2.4.2.

Um diesen Z-Test durchzuführen, wurden in der Publikation zwei verschiedene Ansätze für den Z-Test verglichen. Beim ersten Modell werden beide Vektoren randomisiert, sodass jegliche Zusammenhänge innerhalb des Vektors, aber auch zwischen den Vektoren zerstört werden. Für das zweite Modell werden nur die jeweiligen Ziel-Vektoren randomisiert. Dadurch bleiben intrinsische Korrelationen des Drive („Motors“) erhalten, aber der Bezug zum Zielvektor wird entfernt. Da die Ergebnisse mit beiden Modellen konsistent waren, verwendete ich in dieser Arbeit nur das zuerst erwähnte vollständig randomisierte Modell.

4.2.1 Parallelisierung und Effizienz

Wie eingangs erwähnt, beschäftigte sich die Studie mit einer effizienten Implementierung in C++, da das Erstellen und Füllen der Histogramme zur Berechnung der Transferentropie sehr aufwendig werden kann. Dies ist von verschiedenen Faktoren abhängig: Länge des Vektorpaares, Fenstergröße der Transferentropie und Anzahl der verwendeten Bins.

Der Einfluss der Vektorlänge ist offensichtlich: Die Anzahl der zu berechnenden Paare für x und y steigt dadurch proportional an. Dies bedeutet jedoch nicht, dass die Anzahl der zu befüllenden Histogramme für die Abschätzung der Wahrscheinlichkeiten aus Formel 2.19 im gleichen Maße zunimmt. In der Tat werden die Histogramme in der Regel immer dünner besetzt mit steigendem Fenster m. Das gilt ebenfalls für die randomisierten Histogramme des Z-Tests. Allerdings aufgrund der Tatsache, dass dort die Datenpunkte in einem breiten Ereignisraum verteilt und die Histogramme dadurch immer dünner besetzt werden („Curse of Dimensionality“ [44]).

Abhängig vom verwendeten Datensatz kann sich dadurch der benötigte Speicher drastisch verringern (Im Hinblick auf eine mögliche Parallelisierung auf Grafikprozessoren ein wichtiger Aspekt. Diese, im Englischen als graphics processing unit (GPU) bezeichneten, Prozessoren von Grafikkarten haben sich in in Laufe der letzten Jahre auch für nicht-grafische Berechnungen, die hoch parallel laufen können, etabliert [**? ?**].). Ein weiterer Vorteil hinsichtlich Rechenzeit ist, dass zur Berechnungen nicht immer über alle Einträge mittels verschachtelter Schleifen iteriert werden muss.

Sind die gewünschten Fenstergrößen bereits zur Kompilierungszeit bekannt, lässt sich über Template-Programmierung eine weitere Optimierung vornehmen [15]. Die oben genannten Schlüssel der Map sind faktisch Integerkodierungen von x_t und y_t. Wir erzeugen nun aus m-dimensionalen Schlüsseln $(x_t, x_{t+1}, \ldots, x_{t+m})$ einen einzigen Integerwert. Diese „eindimensionalen“ Schlüssel können nun mittels der Template-Funktionalität von C++ bereits zur Kompilierungszeit erzeugt werden. Zur Kompilierungszeit kann ein bestimmtes Fenster vorgegeben werden. Dadurch wird jedoch die ausführbare Datei größer, sodass abhängig vom Einsatzszenario eine optimale Wahl für m vom Benutzer getroffen werden muss. Ein Vergleich mit verschiedenen Bins ist in Abbildung 4.4 zu sehen. Dort wurden auf einem Desktop PC die Zeiten mit „pre-kompilierten“ Histogrammen und dynamischen Histogrammen verglichen. Zu sehen ist, dass Geschwindigkeitssteigerungen von bis zu 35% möglich sind. Für kurze Vektoren fällt dabei oben gezeigte Steigerung kaum ins Gewicht, wohl aber für längere Datensätze und häufigen Wiederholungen der Berechnung, wie es bei dem Z-Test und der Normierung (Abschnitte 2.4.2 und 2.2.5) der Fall ist.

Der Rechenaufwand wird zusätzlich durch den Z-Test erhöht, da dort die Transferentropie für weitere n randomisierte Vektoren x und y neu berechnet wird. In früheren Arbeiten [118] erwies sich als notwendige Anzahl an Randomisierungsdurchgängen ein Mindestwert von 100 Wiederholungen. Da die Berechnungen jedoch voneinander unabhängig sind, bietet sich hier eine Parallelisierung des Prozesses an. In Abbildung 4.5 zeigen wir die Laufzeit s nach Anzahl der Threads für zwei verschiedene Hardwarearchitekturen. Zur Parallelisierung der Prozesse verwenden wir das OpenMP-Framework, das eine einfache Implementierung gestattet.

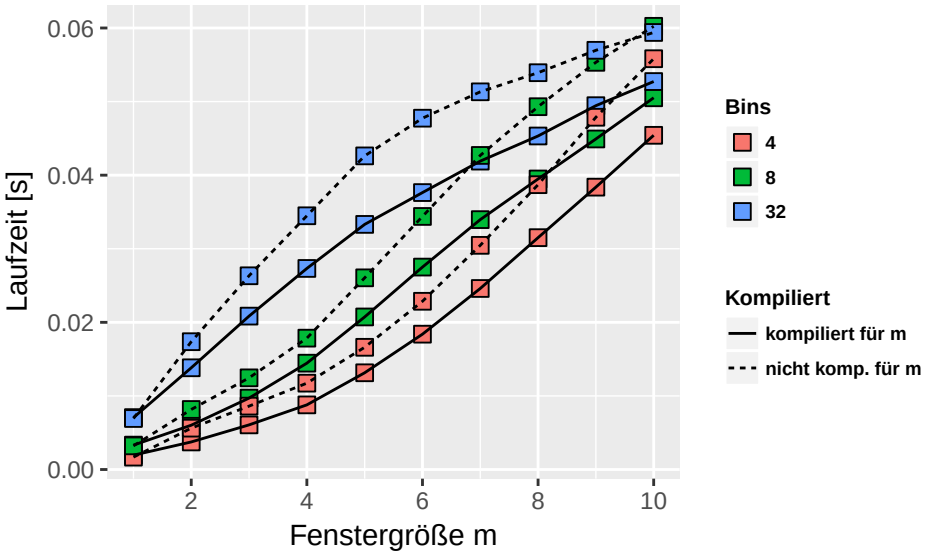


Abbildung 4.4: Laufzeit der Berechnung der Transferentropie für verschiedene Fenstergrößen. Die unterbrochenen Linien deuten an, dass die Ausführung über dynamische indizierte Histogramme erfolgte. Durchgezogene Linien stellen für die angegebenen Größen m kompilierte Histogramme dar. Verglichen wurden vier, acht und 16 Bins bei Fenstergrößen von eins bis zehn. Die Werte sind gemittelt über 1000 unabhängige Berechnungen.

Um die Geschwindigkeitssteigerung zu bewerten, kann man das Amdahlsche Gesetz [3] anwenden. Es beschreibt wie effektiv ein Problem von einem parallelen Berechnungsverfahren gelöst wird. Ein Programm wird demnach nie vollständig parallel laufen können, da bestimmte Teile nicht parallelisierbar sind. Der sequenzielle Anteil der Berechnung wird dadurch mit zunehmender Parallelisierung zum geschwindigkeitsbestimmenden Schritt. Die Geschwindigkeitssteigerung lässt sich somit wie folgt definieren:

$$S[N] = \frac{1}{(1-P) + \frac{P}{N}} \tag{4.1}$$

Dabei ist P der Anteil parallelisierbaren Codes und N die Anzahl der verwendeten CPU-Kerne (bzw. Threads). Für die in Abbildung 4.5 gezeigten Daten ergibt sich anhand der daraus abgeleiteten Geschwindigkeitssteigerung $S[N]$ eine Parallelisierbarkeit P von 97% im Falle der Intel-Architektur und 99% für AMD.

4.3 `R`-Paket: TransferEntropyPT

Auf der Bibliothek aus Abschnitt 4.2 aufbauend wurde von mir ein `R`-Paket entwickelt. Damit wird der Nutzen dieser `C++` Bibliothek, deren Fokus auf effizienter Ausführung liegt, einem breiteren Publikum zu verfügen gestellt, da `R` eine hohe Popularität unter

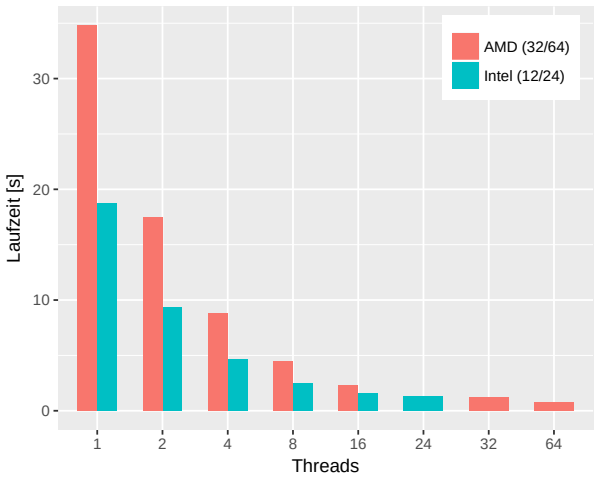


Abbildung 4.5: Laufzeit der Berechnungen auf verschiedene Prozessorarchitekturen (Intel mit 12 Cores/24 Threads und AMD mit 32 Modulen/64 Threads). Deutlich zu erkennen ist die starke Skalierbarkeit mit Anzahl der Threads: Die Laufzeit wird etwa halbiert bei doppelter Anzahl an Threads.

Naturwissenschaftler hat [99]. Indem wir die Bibliothek um ein `R`-Paket ergänzen, können wir sie somit einem breiteren Publikum zur Verfügung stellen. Des weiteren ergänzt es die reine Ermittlung der Transferentropie um weitere Funktionen, wie die Berechnung der theoretischen Unter- und Obergrenzen der Transferentropie. Die Funktionalität des Pakets mit Beispielanwendungen wurde im Rahmen der Konferenz *CMSB – Computational Methods for Systems Biology 2017* in Darmstadt publiziert [17]. Das Listing 8.1 im Anhang zeigt die Ausführung des Pakets. Die Funktion *get.te()* enthält das Binning der Daten (siehe Abschnitt 2.3) als Parameter des Methodenaufrufs.

4.3.1 Normierung der Transferentropie mittels theoretischer Ober- und Untergrenze

Zusätzlich ergänzte ich für diese Studie das Paket um die Möglichkeit eine Normierung (vgl. Gleichung 2.21) der Transferentropie durchzuführen. Diese Funktionalität war in der ersten Version des Pakets noch nicht enthalten, ist aber in der aktuell zum Download angebotenen Versionen 1.1.2 verfügbar. Der Optimierer ist in einen Minimierer und Maximierer geteilt, welche beide nach dem Prinzip eines Greedy-Algorithmus funktionieren. Die beiden Prinzipien werden im Folgenden erläutert.

Der Maximierer bekommt als Eingangsparameter die beiden Zeitreihen, sowie die nötigen Transferentropie Parameter (Bins und Fenster). Für den Optimierungsprozess werden zusätzlich ein Delta d und eine Schrittweite n festgelegt. Diese dienen als Abbruchbedingungen des Greedy-Algorithmus.

In jedem Schritt werden nun innerhalb der beiden Vektoren zufällig Symbole getauscht. Dadurch bleibt die gesamte Komposition eines Vektors unverändert, aber die Reihenfolge der Symbole ändert sich. Ist die anschließend berechnete Transferentropie-Änderung

niedriger als d, wird das erzeugte Vektorpaar verworfen und der Prozess wiederholt. Für diesen Fall werden bis n neue Versuche gestartet. Sollte nach Ablauf der n Versuche kein neues Maximum gefunden worden sein, ist die Optimierung beendet. Wird im Laufe dieser Versuche ein neues Maximum gefunden, wird das erzeugte Vektorpaar gespeichert und dient als Ausgang für den nächsten Zyklus. Das Listing in 4.2 zeigt die Funktionsweise in Pseudo-Code. Der Minimierer funktioniert analog zum Maximierer. Als zusätzliche Abbruchbedingung ist jedoch eine Transferentropie von null integriert, da dies den niedrigsten Wert der Transferentropie darstellt. Jede Optimierung muss zweifach erfolgen. Im ersten Durchgang wird die Transferentropie entsprechend der Richtung $x \rightarrow y$ minimiert und maximiert. Im zweiten Durchgang passiert das Gleiche für die umgekehrte Richtung $y \rightarrow x$. Dadurch entstehen insgesamt für einen normierten Transferentropie-Wert vier zusätzliche Berechnungen. Da diese allerdings unabhängig sind, lassen sie sich sehr gut parallelisieren. Dazu verwendete ich das `R`-Paket doParallel [119].

Auflistung 4.2: Funktionsweise des Optimierers.

```
x       ## 'x' und 'y' sind die beiden Vektoren,
y       ## auf denen die TE maximiert wird.

TE_aktuell = calc_TE(x, y)
Setze i = 0

while (i < n) {

    swap (x)       ## 'swap' tauscht zwei zufllige Elemente in dem
    swap (y)       ## jeweiligen Vektor miteinander aus.

    TE_neu = calc_TE(x, y)

    if ((TE_neu - TE_aktuell) > d ) {
        set TE_aktuell = TE_neu
        set i = 0
    } else {
        set i = i + 1
    }

}
```

5 Ergebnisse

5.1 Anwendbarkeit der Transferentropie

Zuerst ist es wichtig sich einen Eindruck von einem wichtigen Aspekt aller empirischen Studien zu Entropien verschaffen. Nämlich, wie die idealen Histogramme der gemessenen Häufigkeiten aussehen, die als Schätzer für die Wahrscheinlichkeiten in Gleichung 2.19 dienen. Zu diesem Zweck wurden zwei Systeme verwendet, um die Transferentropie hinsichtlich dieses Aspekts zu untersuchen. Zum einen das System einer gekoppelten logistischen Gleichung 3.1 und zum anderen ein Hidden Markov Model (Abschnitt 3.2).

5.1.1 Anwendung auf die Coupled Logistic Map

Zunächst muss festgestellt werden, welche Auflösung (dies reziprok zur Anzahl der Bins) notwendig ist um die kausale Beziehung $x \to y$ richtig zu erfassen und gleichzeitig kein Falsch-Positives Ergebnis für die Richtung $y \to x$ zu erzeugen. Abbildung 5.1 veranschaulicht dies für solch ein gekoppeltes antizipatorisches System (Parameter $\alpha = 1$ und $\xi = 0,4$). Deutlich zu erkennen ist, dass man ein richtiges und unterscheidbares Signal von vier Bins aufwärts bekommt. Deshalb werden in den nachfolgenden Abschnitten, kontinuierlichen Daten mit vier Bins oder mehr diskretisiert bevor auf ihnen die Transferentropie berechnet wird.

Als nächstes stellte sich die Frage nach der statistischen Signifikanz. Zu diesem Zweck wird das in Abschnitt 2.4.2 vorgestellte Z-Score-Modell verwendet. In Falle der Transferentropie lag dabei das Augenmerk darauf, die korrekte Erkennung der Richtung des Informationsflusses zu verbessern. Ein Z-Score sollte also im Falle einer kausalen Beziehung möglichst hoch sein, also Werte größer zwei annehmen, da dies der 2σ Umgebung der Normalverteilung entspricht. Hierbei is zu beachten, dass auch negative Z-Scores auftreten können. Dies kann so interpretiert werden, dass selbst ein vollkommen zufälliger Wert einen höheren Informationsgehalt trägt und damit das Ergebnis als nicht signifikant gewertet werden muss.

Zunächst ist bei der Analyse der Ergebnisse (Abb. 5.2) festzustellen, dass die Transferentropie bei kurzen Vektoren die höchsten Werte liefert. Allerdings widerspricht der Z-Score einer Signifikanz des Ergebnisses. Im Falle des gekoppelten Systems zeigt sich bei einer Fenstergröße von eins sowohl bei einem Binning von vier als auch acht, dass erst eine Vektorlänge von 256 und höher zu einem signifikanten Ergebnis führt.

Für eine Fenstergröße von zwei ist sogar erst bei einer Vektorlänge von 2048 ein signifikantes Signal erkennbar (bins = 4). Bei einem Binning von acht wird Signifikanz anhand des Z-Scores erst bei über zwanzigtausend erreicht (nicht gezeigt). Das bei dieser

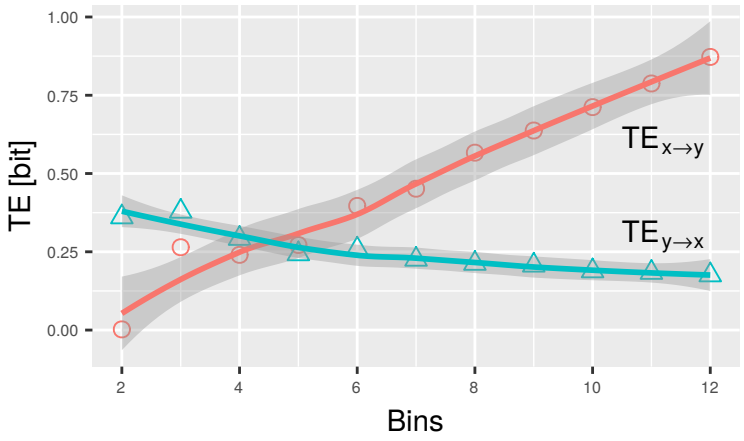


Abbildung 5.1: Gezeigt ist die Transferentropie als Funktion der Anzahl der Bins. Die Trendlinien stellen eine lokale Polynomregressionen dar [26]. Die grauen Bereiche um die Trendlinien sind das Konfidenzintervall (p = 0,95) der Approximation. Der Schnittpunkt der beiden Funktionen liegt bei einem Wert von etwa 4 Bins. Werte die darunter liegen, deuten fälschlicherweise einen umgekehrten kausalen Zusammenhang an. Darüber wird die Richtung des Informationsflusses korrekt erkannt.

Fenstergröße kein eindeutiger Hinweis auf einen Informationsfluss besteht, könnte an der Generierung des Systems liegen. Die interne Schrittweite liegt dort bei eins. Dies könnte bedeuten, dass die Transferentropie fehlschlägt, wenn die Schrittweite durch die gewählte Fenstergröße überschritten wird.

Sind die System unabhängig bleibt der Z-Score erwartungsgemäß niedrig. Für all diese Systeme können keine signifikanten Transferentropien festgestellt werden.

Zusammenfassend kann man festhalten, dass der Z-Score ein probates Mittel darstellt, um die Qualität der Aussage der Transferentropie zu erhöhen: Eine Transferentropie kann zunächst zwar auf einen Informationsfluss hindeuten, aber in diesem Beispielsystem erst als echtes Signal angesehen werden, wenn ebenfalls der Z-Score eine ausreichende Höhe aufweist.

Transferentropie ist, wie die Zeitverzögerte Transinformation 2.2.4, ein effektives informationstheoretisches Maß, wenn Zeitreihen von Variablen mit diskreten Ereignissen erzeugt werden. Aber auch ohne diese Vorraussetzung kann mittels geschickt gewähltem Binning der Ausgangsdaten die Transferentropie vielseitig eingesetzt werden, wie in Kapitel 5 gezeigt wurde.

Um die Stärke und Aussagekraft des reinen Werts zu beurteilen, muss die Signifikanz davon festgestellt werden. In Abschnitt 5.1 konnte gezeigt werden, dass die Z-Score-Berechnung als Methode zur Feststellung der Signifikanz des Ergebnis dienen kann. Anhand zweier Testsysteme (In Abschnitt 5.1.1 die Couple Logistic Map und in 5.1.2 ein Hidden Markov Model), deren Stärke des kausalen Zusammenhangs kontrolliert werden kann, wie der Z-Score hilfreich bei der Beurteilung der Signifikanz des Ergebnisses ist. Mittels diesen Ansatzes konnten wir zeigen, dass ein signifikantes Ergebnis der Transfe-

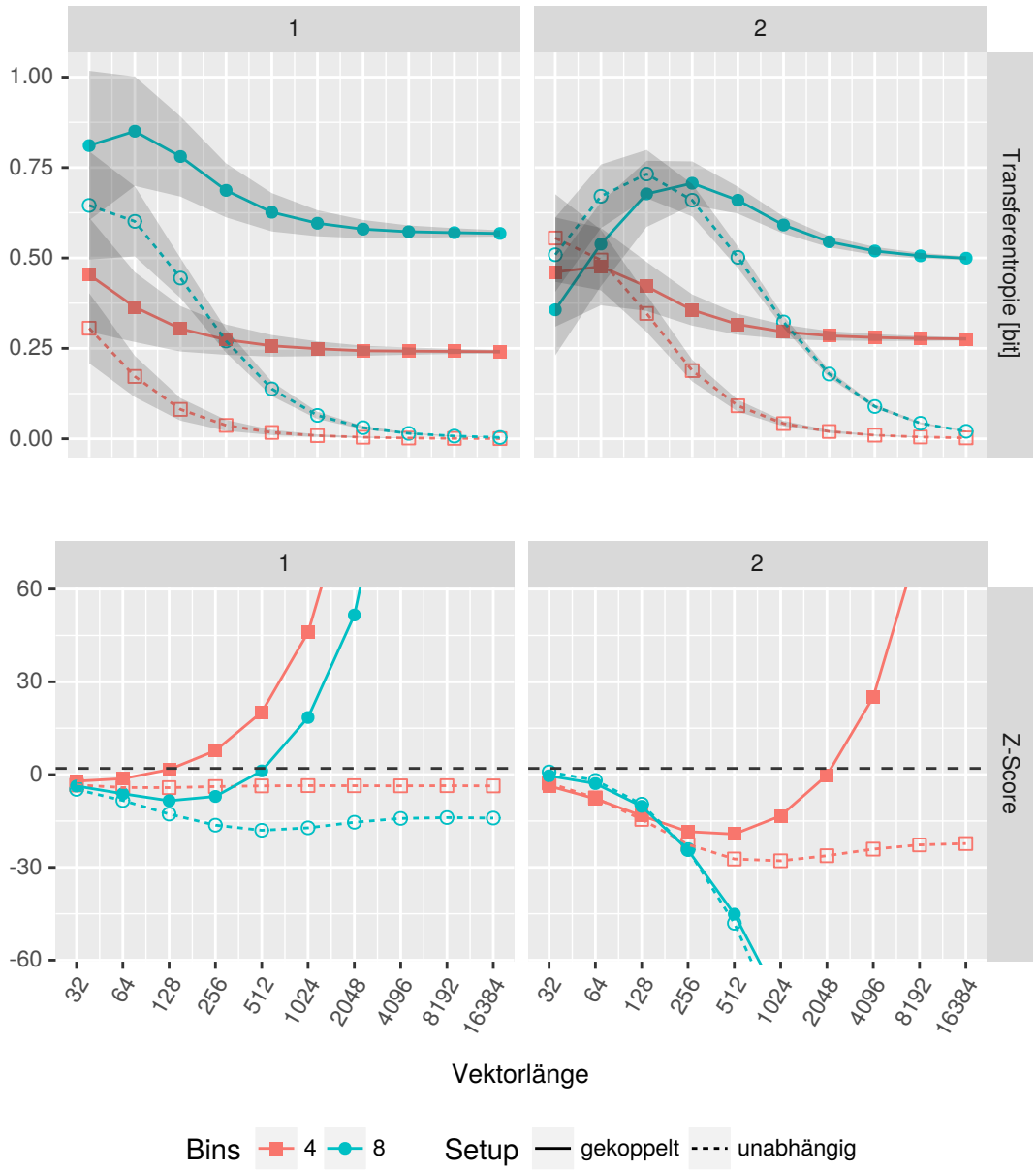


Abbildung 5.2: Die Grafik zeigt die Transferentropie (oben), bzw. deren Z-Scores (unten) in Abhängigkeit von der Größe der Daten für das System der Coupled Logistic Map. Die Schattierungen stellen die Standardabweichung der 1000 wiederholten Berechnung pro Datenpunkt dar. Die durchgezogenen Linien sind Systeme mit tatsächlichem Informationsfluss ($x \to y$), die gestrichelten Linien sind dagegen unabhängige Systeme. Auf der linken Seite der Grafik befindet sich die Auswertung für die Fenstergröße von $m = 1$ und auf der rechten für $m = 2$. Während sich bei vier Bins (rot) bereits ein Z-Score von zwei ab Vektorlänge 128 einstellt, benötigt man für acht (grün) und 16 Bins mindestens 512, respektive 2048 Datenpunkte (für $m = 1$). (Anmerkung: Da die y-Achse auf +/- 60 begrenzt ist, werden in den unteren Abbildungen höhere und niedrigere Werte nicht dargestellt.)

rentropie auch für kleine Datensätze berechnet werden kann.

5.1.2 Anwendung auf ein Hidden Markov Model

Als nächstes folgt die Untersuchung des Hidden Markov Models aus Abschnitt 3.2 und inwiefern die Z-Scores bei der Beurteilung des Ergebnisses helfen können. Abbildung 5.3 zeigt die Transferentropie und die Z-Scores der beiden genannten Modelle. Erkennbar ist zunächst, dass die Transferentropie für die Richtung $x \to y$ stets höher ist als in die entgegengesetzte Richtung (Abb. 5.3a). Dies gilt für beide Modelle, sodass anzunehmen ist, dass die interne Komplexität des Systems für die Erkennung des Informationsflusses unerheblich ist.

Dennoch kann der Z-Score zur Unterstützung heran gezogen werden, denn schließlich ist auch hier noch unklar, ob der Unterschied der beiden Signale auch signifikant ist. Abbildung 5.3b zeigt eindeutig, dass nur in Richtung $x \to y$ ein signifikanter Informationstransfer statt findet, da die Z-Scores sich um mehrere Größenordnungen unterscheiden. Schon bei kleinen Datensätzen (ab 32 Datenpunkten) ist dieser Unterschied deutlich feststellbar und nimmt mit zunehmendem Umfang der Daten weiter zu.

Dieses Ergebnis unterstreicht ebenso wie in Abschnitt 5.1.1 den Nutzen des Z-Score-Referenzmodells. Die Scores steigen monoton mit der Datensatzgröße an. Ein Effekt der durchaus zu erwarten ist, da mit umfangreicherer Stichprobe, üblicherweise die Qualität selbiger ansteigt.

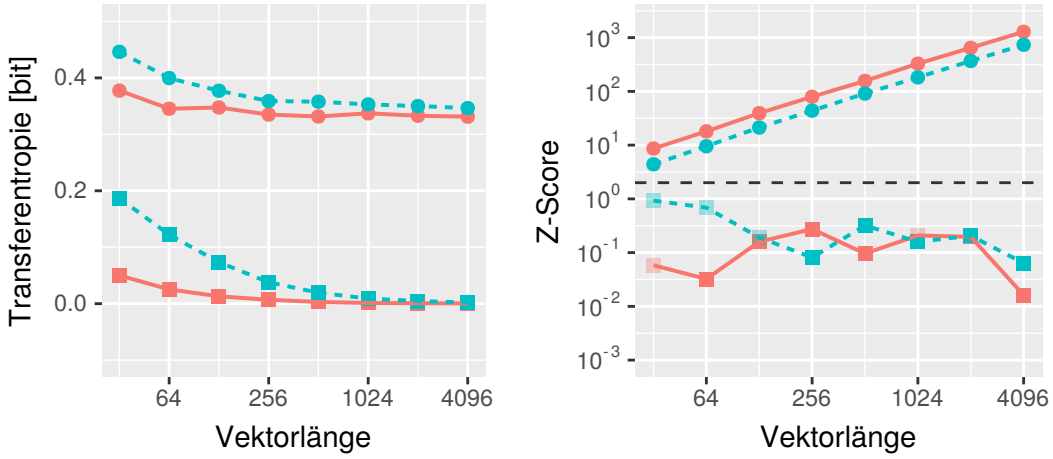
5.1.3 Fazit zur Anwendbarkeit der Transferentropie

Wie in den vorigen Abschnitten gezeigt, kann die Transferentropie als verlässliches Maß zur Erkennung des Informationsflusses zwischen zwei Vektoren verwendet werden. Randbedingungen sind dabei vor allem die verwendeten Parameter. Die gezeigten Modellsysteme erlauben eine Einschätzung, da sie einen gerichteten Informationsfluss zwischen zwei Vektoren simulieren.

Um die Verlässlichkeit der Aussage zu erhöhen, kann der Z-Score unterstützend eingesetzt werden. Allerdings kann der Z-Score abhängig durch die hohe Anzahl an Shufflings und neuen Berechnungen die Laufzeit für eine Auswertung deutlich erhöhen. Daher wurden Optimierungen des Codes nötig, um die umfangreichen Auswertungen der biologischen Netzwerkmodelle zu ermöglichen. Abschnitt 4.2.1 beschreibt die implementierten Verbesserungen zur Effizienzsteigerung der Berechnungen.

5.2 Netzwerkrekonstruktion

Die Auswertung der Netzwerkrekonstruktion erfolgte an verschiedenen Systemen. Diese werden in den folgenden Abschnitten beschrieben. Zusätzlich zu den fünf verschiedenen Maßen wurden für alle Maße ein Signifikanzwert ermittelt. Für Transferentropie, zeitverzögerte Transinformation und die Informationsflussrate war dies der Z-Score. Für die



(a) (b)

Abbildung 5.3: (a) zeigt die Transferentropie der zwei verwendeten Hidden Markov Modelle für unterschiedliche Vektorlängen (Anzahl der Emissionen). In rot ist das erste Modell und in blau das zweite Modell dargestellt. Die Übergangswahrscheinlichkeiten für die Modelle sind in den Tabellen 3.1 und 3.2 zu finden. In der logarithmischen Auftragung in Abbildung (b) sind negative Werte als ihr Betrag und transparent dargestellt. Die gestrichelte Linie befindet deutet einen Werte von zwei an.

Kreuzkorrelation wurde das Konfidenzintervall von 95% berechnet und für die Korrelation wurde ein p-Wert von 0,05 gewählt. Die Ergebnisse der verschiedenen Maße wurden dann anhand dieser Kriterien gefiltert, sodass diese Werte aus der Berechnung der VUS entfernt wurden.

5.2.1 Einfaches Differentialgleichungssystem

Für die Analyse des DGL-Systems (siehe Abschnitt 3.3) wurden für vier verschiedene Konstanten k die Differentialgleichungen für $n = 10^6$ Schritte simuliert. Für die Kopplungskonstante k wurden vier verschiedene Werte zwischen null und zehn eingesetzt, die die Stärke der Kopplung reflektieren. Für die Werte $k = 0$ und $k = 0,001$ ist eine geringe Kopplung zu erwarten, während $k = 1$ und $k = 10$ eine auffällige Kopplung und Einfluss der beiden Zyklen zeigen sollte.

Von dieser umfangreichen Simulation wurden anschließend Stichproben in regelmäßigen Abständen entnommen, sodass Datensätze der Länge 30, 50, 250 und 500 entstanden. Insgesamt wurden von diesen Stichproben jeweils hundert entnommen, sodass die im folgenden gezeigten Daten eine Mittlung dieser Stichproben sind.

Die in Abschnitt 2 vorgestellten Maße wurden auf diesen Datensätzen berechnet und zu allen einhundert Berechnungen die Fläche unter der Kurve für die Sensitivität (Richtig-positiv-Rate) gegen die Ausfallrate (Falsch-positiv-Rate). Dabei wurden die in Tabelle 5.1 gezeigten Adjazenzmatrizen als Referenz für die ROC verwendet.

	u→	v→	x→	y→
→u	0	1	0	0
→v	1	0	0	0
→x	0	0	0	1
→y	0	0	1	0

(a) Ungekoppeltes System

	u→	v→	x→	y→
→u	0	1	0	0
→v	1	0	0	0
→x	0	0	0	1
→y	0	1	1	0

(b) Gekoppeltes System

Tabelle 5.1: Adjazenzmatrizen des einfachen Differentialgleichungssystems. Das grau unterlegte Feld in den Tabellen hebt den einzigen Unterschied zwischen den beiden Adajazenzmatrizen hervor.

Die Parametrisierung der Maße erfolgte für alle gleichermaßen mit einem Zeitfenster von $t = 1$. Zusätzlich wurden für Transferentropie (TE), Informationsflussrate (RIF) und zeitverzögerte Transinformation (TDMI) noch Z-scores berechnet. Für die Transferentropie wurde außerdem der in Abschnitt 2.2.5 vorgestellte Ansatz zur Normierung berechnet. Sowohl Transferentropie als auch Transinformation verwenden ein Binning von acht. Abbildung 5.4 zeigt das Ergebnis der Auswertung.

Um die Ergebnisse der VUS Berechnung besser einordnen zu können wurde die gleiche Auswertung mit einer „falschen“ Adjazenzmatrix durchgeführt. Diese wird im Folgenden als Nullmodell bezeichnet. Die dazu verwendeten Matrizen enthalten die gleiche Anzahl an Kanten, jedoch werden diese randomisiert. Die Diagonale der Matrix bleibt dabei allerdings unberührt, sodass sich die in Tabelle 5.2 folgenden Matrizen ergeben.

	u→	v→	x→	y→
→u	0	0	1	0
→v	0	0	1	0
→x	0	1	0	0
→y	1	0	0	0

(a) Ungekoppeltes System

	u→	v→	x→	y→
→u	0	1	1	0
→v	0	0	1	0
→x	0	1	0	0
→y	1	0	0	0

(b) Gekoppeltes System

Tabelle 5.2: Adjazenzmatrizen des Nullmodells des Differentialgleichungssystems. Die grau unterlegten Felder entsprechen den Kanten die in den richtigen Systemen vorhanden sind.

Zunächst fällt auf, dass die VUS der Z-Scores der VUS des jeweiligen Maßes direkt entspricht. Das bedeutet, dass ein hoher Z-Score direkt mit einem vorhergesagten Kontakt im Zusammenhang steht. Ebenfalls weißt der normierte Wert der Transferentropie eine ähnliche VUS auf wie Rohwerte oder Z-Score der Transferentropie.

Für eine größere Stichprobe wäre eine höhere VUS zu erwarten gewesen. Dieses Verhalten wird auch von der zeitverzögerten Korrelation, Kreuzkorrelationsfunktion und der Informationsflussrate weitgehend bestätigt. Allerdings zeigen die zeitverzögerte Transinformation und vor allem die Transferentropie ein gegenteiligen Trend. Dies könnte im Zusammenhang mit den Ergebnissen in Abbildung 5.2a stehen. Auch hier war zu erkennen, dass erst ab einer Vektorgröße von 500 die Transferentropie konstante Ergebnisse erzielte. Zusammenhängen könnte dieses Phänomen mit den dünn besetzen Histogrammen der Häufigkeiten, auf denen die Maße berechnet werden.

Am stabilsten hinsichtlich der unterschiedlichen Parametrisierung der Parameter stellt sich die Transferentropie dar. In allen vier Szenarien schnitt sie vergleichbar gut ab (VUS von ca. 0,75). Interessant ist die Transinformation unter diesem Aspekt. Für die ungekoppelten Systeme liegt der VUS bei (nahezu) eins. Damit würde das Maß die perfekte Rekonstruktion der Adjazenzmatrix bedeuten. Allerdings kehrt sich dies ins Gegenteil für die gekoppelten Systeme. Hier schneidet das die Transinformation mit einem VUS von 0,25 bis 0,5 deutlich schlechter ab.

5.2.2 Markov-Ketten Netzwerke

Im Folgenden unterziehe ich die von mir in Abschnitt 3.4 vorgestellten Markov-Netzwerke (Rényi- und Barabási-Modell) der Analyse mit den verschiedene Maßen. Für die Auswertung wurden Variationen der in dem Abschnitt erläuterten Parameter erzeugt, um ausreichende Variabilität der Simulationen zu erhalten. Eine tabellarische Auflistung der Parameter für die Übergangswahrscheinlichkeiten befindet sich im Anhang 8.1 und die Flipwahrscheinlichkeiten wurden auf $0,6$, $0,7$ oder $0,8$ gesetzt.

Für die Auswertung wurden sowohl für Rényi- als auch Barabási-Netzwerke verschiedene Kombinationen für den zeitlichen Versatz (Zeitfenster gleich eins oder zwei) und, im Falle der Entropie basierten Maße, der Bins (vier oder acht) gewählt. Die Ergebnisse verhielten sich für jeweils beide Netzwerktypen sehr ähnlich, daher sind in den beiden

folgenden Abschnitten nur die Ergebnisse für das Szenario der VUS Berechnung für ein Zeitfenster von eins und ein Binning von acht gezeigt.

Um eine bessere Aussage über die Qualität der verschiedene Maße treffen zu können, wurde wie im Abschnitt 5.2.1 zum Vergleich die Auswertung mittels VUS zusätzlich mit einer falschen Adjazenzmatrix durchgeführt (Nullmodell). Diese randomisierte Adjazenzmatrix behält die Anzahl an Kanten bei, verteilt sie aber neu auf der Matrix (ausgenommen Diagonale).

Rényi-Netzwerke

Die diesem Abschnitt diskutierten Ergebnisse beziehen sich auf das Rényi-Modell. Die Anzahl der Eingangskanten die solch ein Graph haben kann, liegt bei maximal drei und es gibt keine isolierten Knoten. Die Zeitreihen wurden mit einer unterschiedlicher Anzahl an Schritten simuliert (Vektorlängen: 25, 50, 250 und 500).

Außer der Cross-Correlation-Function und vor allem der Informationsflussrate zeigen die anderen Maße (Cor, TDMI, TE) einen hohen Grad der Rekonstruktion der Netzwerke. Sowohl Transferentropie, als auch die zeitverzögerte Transinformation und Korrelation zeigen eine gute Erkennung der Netzwerke. Auffällig ist insgesamt die Tatsache, dass die nach Z-Score gefilterten Maße alle deutlich in der VUS abfallen, und somit schlechter geeignet sind die Netzwerkstruktur wiederzuerkennen. Wird lediglich der Z-Score anstelle des Maßes eingesetzt, unterscheidet sich das Ergebnis kaum.

Hinsichtlich des Nullmodells zeigt, wie zu erwarten, keins der Maße eine hohe VUS. Die maximalen Werte von 0,5 deuten an, dass es sich um reine Zuvallsverteilung hinsichtlich der Treffsicherheit des jeweiligen Maßes handelt. Damit lässt sich um Umkehrschluss annehmen, dass die Berechnungen auf den passenden Adjazenzmatrizen verlässlich sind.

Barabási-Netzwerke

Der gleichen Analyse aus dem vorigen Abschnitt wurde auch das Barabási-Modell unterzogen. Die Ergebnisse sind der Abbildung 5.7 zu entnehmen. Die Struktur dieser Netze unterscheidet deutlich von den der Rényi-Netzwerke. Während diese einen eher linearen Aufbau haben mit wenigen Eingangskanten, haben Barabási-Netzwerke typischerweise wenige stark vernetze Knoten und viele mit nur einer oder zwei Kanten. Auch für diese Netzwerke wurden die oben genannten Parameter (Tabelle 8.1) und Flipwahrscheinlichkeiten verwendet.

Hier zeichnet sich im Gegensatz zum vorigen Abschnitt ein anderes Bild. Mit Ausnahme der Transferentropie zeigt, kein Maß ein korrekte Erkennung der Netzwerkstruktur. Und selbst die Ergebnisse der Transferentropie sind nur bei einer Vektorlänge über 50 stabil in einem Bereich jenseits von 0,5.

Das Nullmodell birgt keine Überraschungen. Wie bereits bei den Rényi-Netzwerken kommen bei keinem der verwendeten Maße Werte über 0,5 zustande, sodass auch hier die Methode des VUS als zuverlässige Methode zur Einschätzung der Qualität der verschieden Maße gelten kann.

5.2.3 Biologische Netzwerke aus der BioModels Database

Um ein umfangreiches Bild der einzelnen Netzwerkmaße zu bekommen, wurden verschiedene Stichproben aus den Simulationen entnommen. Um die Relevanz der Stichprobengröße im Zusammenhang mit der Rekonstruktion der Netzwerke zu betrachten, wurden Stichproben mit einer Größe von 25, 50 und 250 Datenpunkten entnommen.Die Größe 500 zeigte in den vorigen Abschnitten keinen Unterschied zum Stichprobenumfang 250 und kann daher vernachlässigt werden.

Aus der Simulation ohne Rauschen wurden für jede Stichprobengröße jeweils 25 Zeitreihen entnommen. Für die Simulationen mit Rauschen wurden jeweils 25 verschiedene Simulationen gestartet. Aus diesen wurden dann ebenfalls für alle Stichprobengrößen je 25 Zeitreihen gezogen. Für die Berechnungen der Netzwerkmaße wurde das Zeitfenster auf eins oder zwei gesetzt. Die Entropie-basierten Maße wurden außerdem einmal mit einer Bingröße von acht und 16 berechnet.

Abbildung 5.8a zeigt das Ergebnis der Auswertung für eine Zeitfenstergröße von eins und einer Bingröße von acht für das Netzwerk BM12. Praktisch alle Netzwerkmaße können die Netzwerkstruktur nicht rekonstruieren. Die Ausnahmen sind TDMI und RIF. Während letzteres für alle Stichprobengrößen eine VUS von ca. 0.75 ergibt zeigt sich bei der TDMI ein ähnlich gutes Signal nur für die Stichprobengröße 250.

In Abbildung 5.8 sind die Ergebnisse für die Netzwerke 95, 99, 106 und 160 gezeigt.

Wie Abbildung 5.8b zu entnehmen ist, kann die Transferentropie die Netzwerkstruktur nicht rekonstruieren. Auf einen ROC VUS Wert von 0,75 schaffen es die Varianz-basierten Maße (Correlation, RIF und Kreuzkorrelation), selbst bei einer geringen Stichprobengröße von 25. Die TDMI schafft diesen Wert nur mit einer Größe von 250. Bei geringerem Probenumfang fällt die VUS erkennbar ab. Auffällig ist, dass die Z-Scores der RIF deutlich schlechtere Resultate bringen, als der Rohwert selbst.

Für das Netzwerk BM99 (Abbildung 5.8c) zeichnet sich ein ähnliches Bild, wie zuvor für BM95. Die Varianz-basierten Maße bieten die besten Ergebnisse im Vergleich. Jedoch ist ein wesentlicher Unterschied, dass die Ergebnisse alle keinen VUS über 0,6 zeigen. Dadurch kann man fest halten, dass keins der Maße eine gute Rekonstruktion des Netzwerks zulässt. Auch für Netzwerk 106 (Abbildung 5.8d) ist eine Rekonstruktion nicht möglich. Keins der Maße (also weder Entropie- noch Varianz-basiert) erreicht eine höhere VUS als 0,5.

Das letzte Netzwerk (Abbildung 5.8e, BM160) bietet ein gemischtes Bild. Die RIF zeigt auffällig niedrige Werte bei nahezu null. Die meisten anderen Maße erreichen auch nicht mehr als 0.55, nur die Kreuzkorrelation kommt auf etwa 0.65. Dadurch lässt sich auch in diesem Fall mit keiner der Methoden eine gute Netzwerkrekonstruktion durchführen.

5.2.4 Verteilung der Werte

Nicht nur der Mittelwert der einzelnen VUS Werte spielt eine Rolle. Auch die Verteilung im Ergebnisraum ist wichtig, um eine Aussage über die Verlässlichkeit der Maße zu bekommen. Für das Rényi Netzwerk und das BioModels Netzwerk BM95 sind solche Verteilungen exemplarisch in Abbildung 5.9 gezeigt.

Betrachtet man die Verteilung der Werte für das Rényi Netzwerk (Stichprobengröße 50) in Abbildung 5.9a, zeigen die Kreuzkorrelation und die Informationsflussrate eine deutliche Akkumulation bei einem Wert von 0,5. Das bedeutet, es findet keine verlässlich Rekonstruktion statt, sondern die Ergebnisse sind praktisch zufällig. Auch für die Transferentropie fallen viele Werte in diesen Bereich. Jedoch gibt es eine zweite Population, die ihren Peak etwa bei 0,75 hat, sodass hier durchaus eine Erkennung der Netzwerkstruktur erfolgt. Am besten funktioniert die Rekonstruktion per zeitverzögerter Korrelation, aber auch mit der zeitverzögerten Transinformation. Erstere aggregiert die meisten Werte im Bereich bei eins, was einer vollständigen Rekonstruktion des jeweiligen zugrunde liegenden Netzwerks entspricht. Im direkten Vergleich dazu befinden anteilig mehr Werte der TDMI unter 0,75.

Sieht man die entsprechende Grafik für eine Stichprobengröße von 250 in Abbildung 5.9b, erkennt man, dass die Verteilung der Werte für alle fünf Maße stärker in Richtung eins verschoben ist. Aber auch hier zeigen sich weitestgehend die gleichen Verhältnisse. Die Kreuzkorrelation und Informationsflussrate schneiden am schlechtesten ab, da sich hier auch wieder die meisten Werte im Bereich des Zufalls bewegen. Die Transferentropie scheint aber von den längeren Vektoren zu profitieren: Ein Teil der Werte liegt hier im Bereich von eins, zeigt also eine optimale Rekonstruktion des Netzwerks. Die TDMI und Korrelation zeigen noch mehr Werte im hohen Bereich. Beide eignen sich in diesem Szenario am besten um die Struktur des Netzwerks aufzuklären.

In der unteren Reihe befindet sich links die Verteilungsdichte der Ergebnisse für das Netzwerk BM95 bei einer Stichprobengröße von 50 (Abb. 5.9c). Hier zeigt sich vor allem die Kreuzkorrelation als geeignetes Maß zur Erkennung des zugrunde liegenden Netzwerks. Nahezu alle Werte liegen bei der CCF im Bereich zwischen 0.8 und 0.85. Damit ist in diesem Anwendungsfall deutlich besser als die restlichen Methoden. Nur die Informationsflussrate weißt vergleichbar hohe Werte auf, aber insgesamt sind die Werte deutlich weiter verteilt, sodass auch hier nur unter bestimmten Voraussetzungen die Netzwerkstruktur gut rekonstruiert wird. Die Korrelation verhält sich ähnlich mit insgesamt leicht niedrigeren Werten (zwei Populationen mit Peaks bei 0.72 und 0.77). Die TDMI fällt danach weiter ab mit gleichmäßig verteilten Werten zwischen 0.4 und 0.7. Die Transferentropie kommt nicht über eine VUS von 0.3 hinaus und erlaubt somit keinerlei Rekonstruktion des Netzwerks.

Auch in diesem Fall führt eine größere Anzahl an Schritten (250, Abbildung 5.9d) der Zeitreihen zu einer Verbesserung des Gesamtergebnisses, insgesamt bleibt das Bild aus der Untersuchung mit 50 Schritten aber erhalten. Einzig die TDMI zeigt hier ein stabileres Resultat bei einer guten Erkennung (VUS bei aggregiert ca. 0.75). Die Transferentropie ist auch hier nicht in der Lage das Netzwerk zu rekonstruieren.

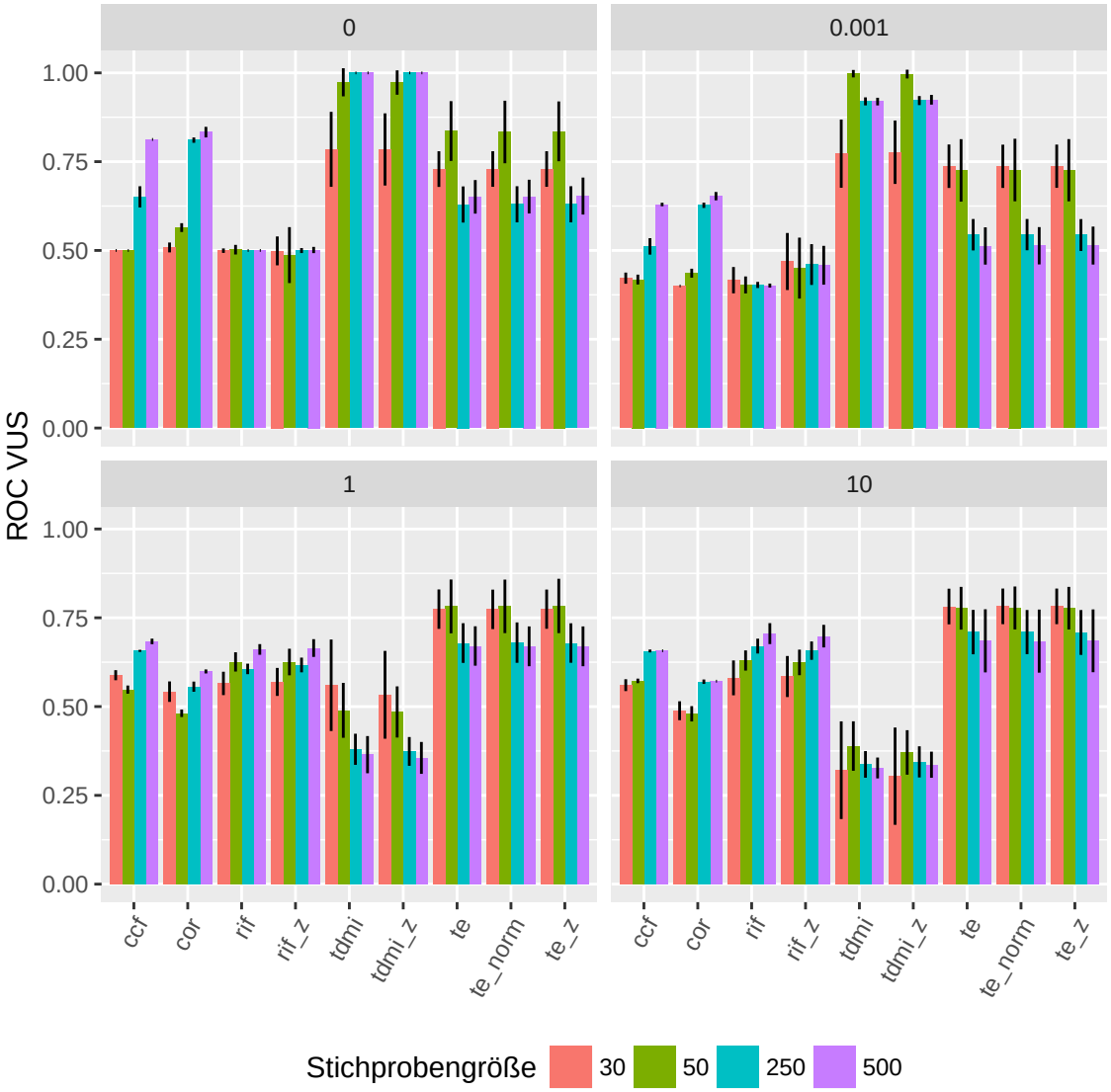


Abbildung 5.4: Die für hundert Messungen (Fehlerbalken sind die Standardabweichung) gemittelte VUS der verschiedene Maße für das DGL System. Gezeigt sind die Ergebnisse für vier verschieden Kopplungskonstanten k (0, 0,001, 1, 10). Alle Maße wurden mit einem zeitlichen Versatz von $l + 1$ parametrisiert. Die obere Reihe verwendet die Adjazenzmatrix 5.2a, die untere Reihe 5.2b. Die gezeigten Maße sind: Kreuzkorrelation (ccf), Korrelationskoeffizient (cor), Informationsflussrate (rif), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), normierte TE (te_norm), Z-Score der TE (te_z).

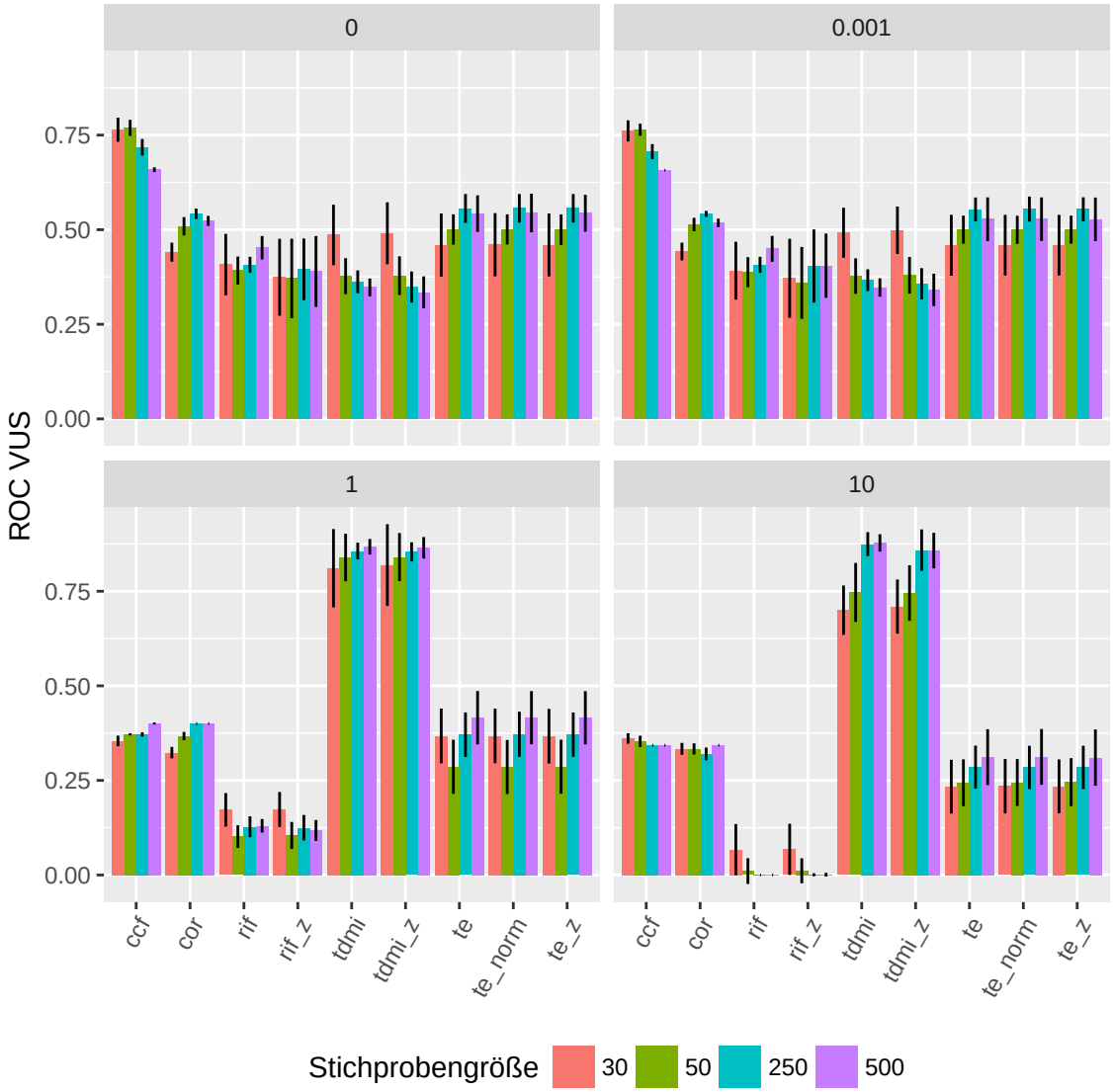


Abbildung 5.5: Für hundert Messungen (Fehlerbalken sind die Standardabweichung) gemittelte VUS der verschiedene Maße für das DGL System. Allerdings wurden hier die ROCs mit im Vergleich zu Abbildung 5.4 randomisierten Adjazenzmatrizen (Nullmodell) berechnet. Gezeigt sind die Ergebnisse für vier verschieden Kopplungskonstanten k (0, 0,001, 1, 10). Alle Maße wurden mit einem zeitlichen Versatz von $l + 1$ parametrisiert. Die obere Reihe verwendet die Adjazenzmatrix 5.3b, die untere Reihe 5.3a. Die gezeigten Maße sind: Kreuzkorrelation (ccf), Korrelationskoeffizient (cor), Informationsflussrate (rif), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), normierte TE (te_norm), Z-Score der TE (te_z).

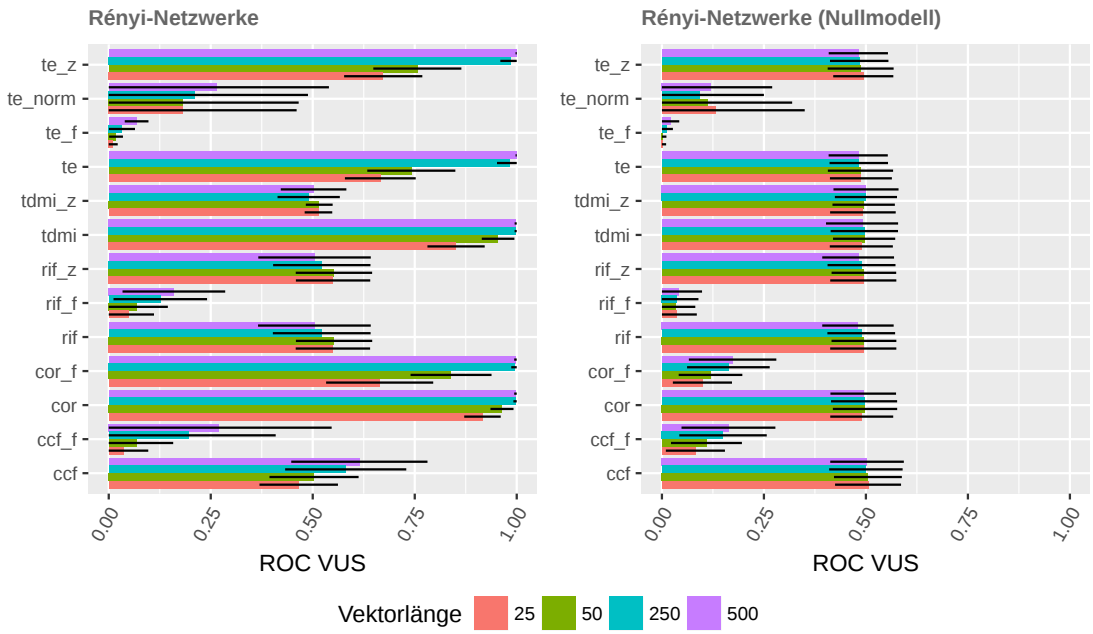


Abbildung 5.6: Berechnung der VUS für das Rényi-Modell. Die gezeigten Werte sind Mittelwerte von 750 Berechnungen, die Fehlerbalken die Standardabweichung. Die einzelnen Maße wurden des weiteren unterschiedlich aufbereitet, sodass für einige Maße deren Z-Score benutzt wurde (_z) um die VUS zu berechnen. Das Suffix _f bedeutet, dass das Maß anhand des jeweils ausgewählten Signifikanzkriteriums gefiltert wurde. Für Die Berechnung erfolgte für vier verschieden Vektorlängen der Zeitreihen (25, 50, 250 und 500). In der linken Abbildung wurden die korrekte Adjazenzmatrizen als Referenz für die VUS verwendet, in der rechten das Nullmodell (randomisierte Adjazenzmatrix). Die gezeigten Maße sind: Kreuzkorrelation (ccf), gefilterter Kreuzkorrelation (ccf_f), Korrelationskoeffizient (cor), gefilterter Korrelationskoeffizient (cor_f), Informationsflussrate (rif), gefilterte Informationsflussrate (rif_f), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), gefilterter TE (te_f), normierte TE (te_norm), Z-Score der TE (te_z).

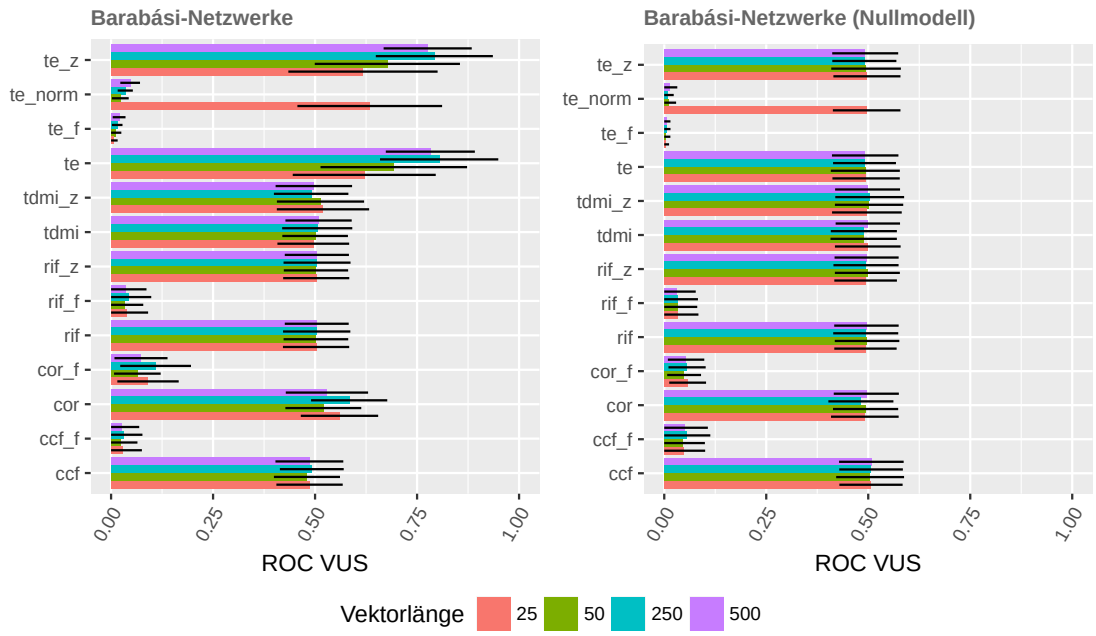


Abbildung 5.7: Berechnung der VUS für das Barabási-Modell. Die gezeigten Werte sind Mittelwerte von 750 Berechnungen, die Fehlerbalken die Standardabweichung. Die einzelnen Maße wurden des weiteren unterschiedlich aufbereitet, sodass für einige Maße deren Z-Score benutzt wurde (_z) um die VUS zu berechnen, bzw. anhand dessen die Rohdaten gefiltert wurden (_f). Die Berechnung erfolgte für vier verschieden Vektorlängen der Zeitreihen (25, 50, 250 und 500). In der linken Abbildung wurden die korrekte Adjazenzmatrizen als Referenz für die VUS verwendet, in der rechten das Nullmodell (randomisierte Adjazenzmatrix). Die gezeigten Maße sind: Kreuzkorrelation (ccf), gefilterter Kreuzkorrelation (ccf_f), Korrelationskoeffizient (cor), gefilterter Korrelationskoeffizient (cor_f), Informationsflussrate (rif), gefilterte Informationsflussrate (rif_f), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), gefilterter TE (te_f), normierte TE (te_norm), Z-Score der TE (te_z).

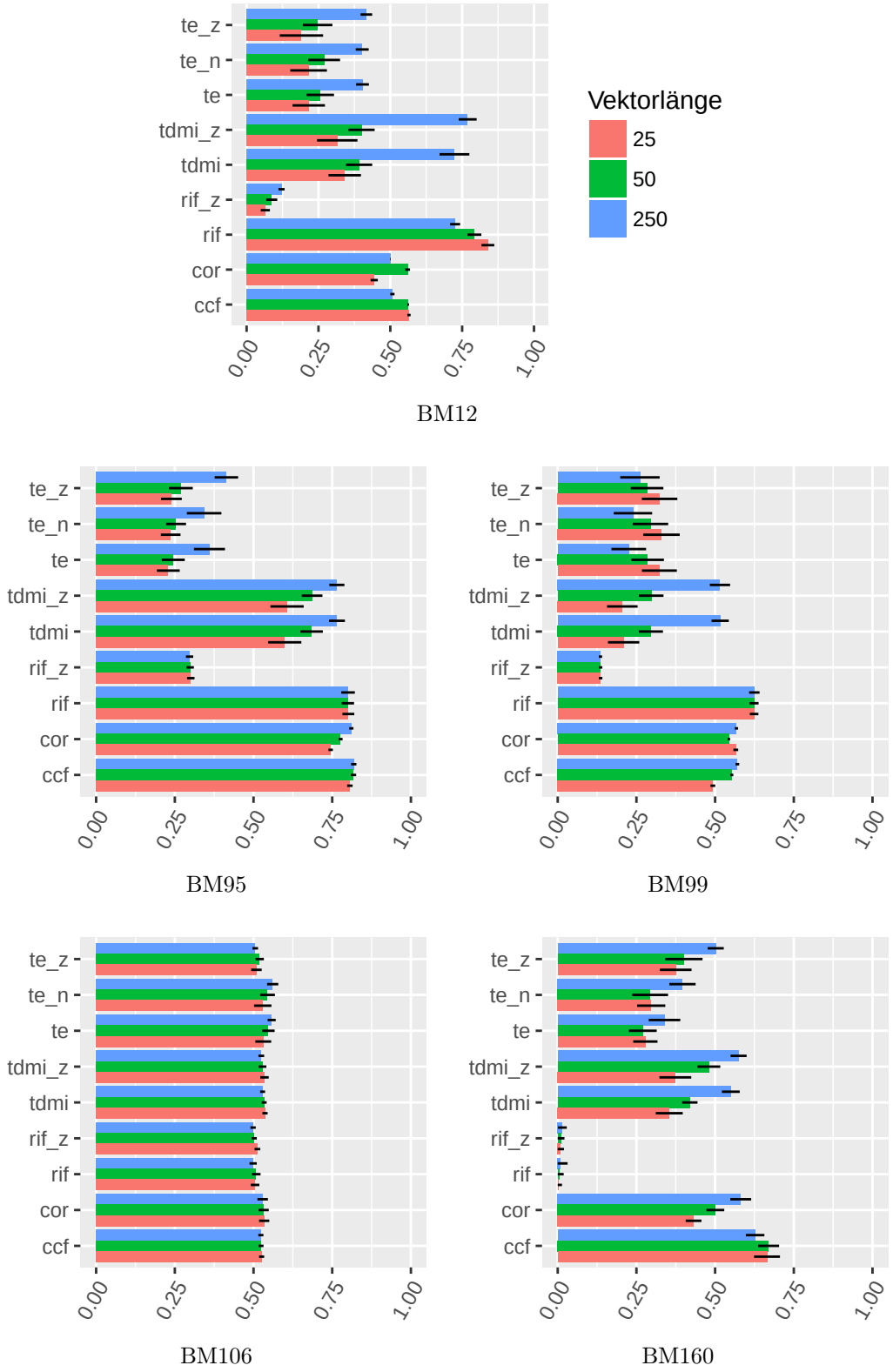
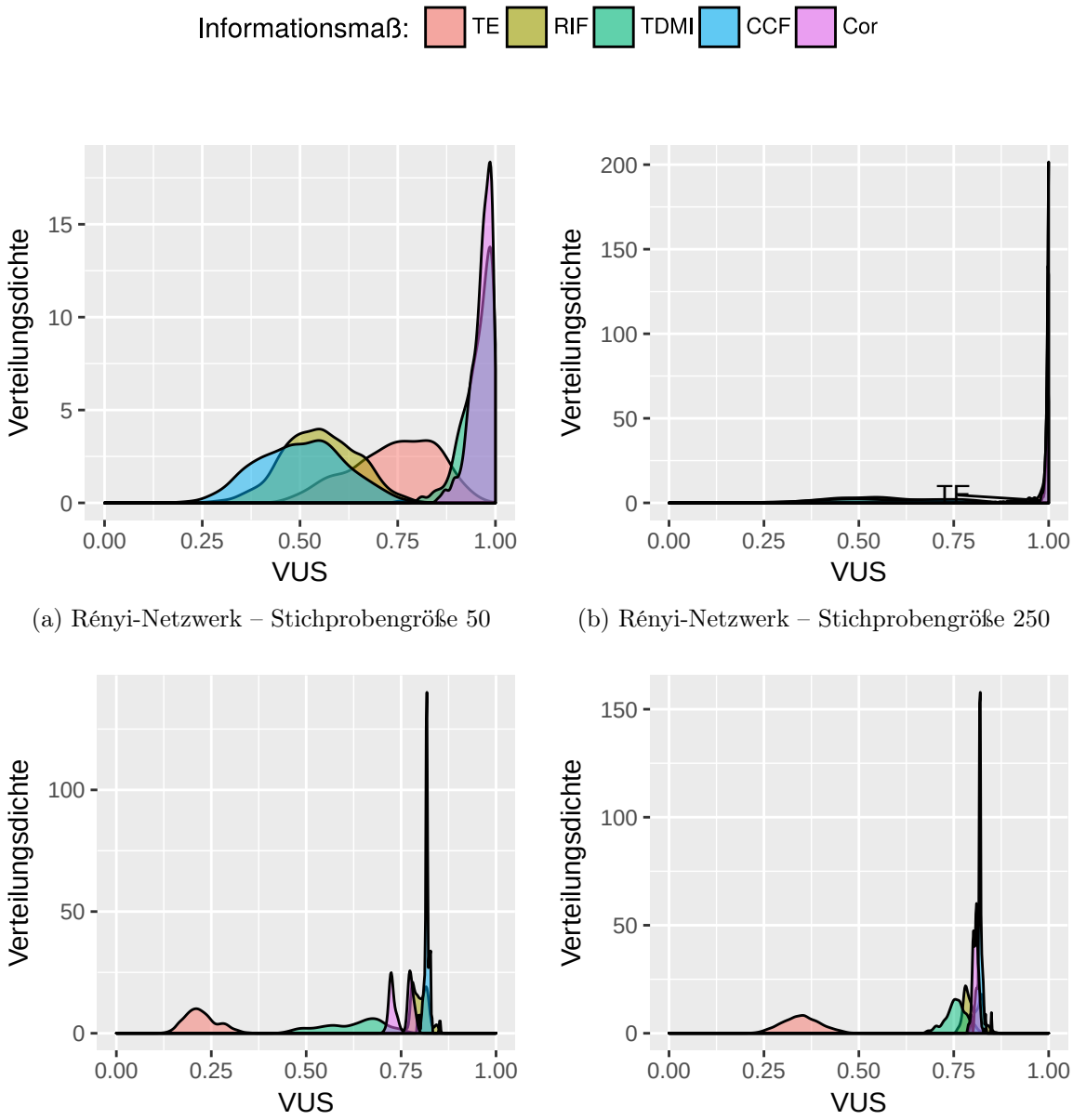


Abbildung 5.8: VUS für die BioModels-Netzwerke für ein Zeitfenster von eins bei einem Binning von acht. Die Balken stellen den Mittelwert aus den einzelnen Berechnungen des VUS für die verschieden Parameterkombinationen dar (Fehlerbalken sind die Standardabweichung). Normierte Transferentropien sind mit _n und der Z-Score mit _z gekennzeichnet. Die gezeigten Maße sind demnach: Kreuzkorrelation (ccf), Korrelationskoeffizient (cor), Informationsflussrate (rif), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), normierte TE (te_n), Z-Score der TE (te_z).



(a) Rényi-Netzwerk – Stichprobengröße 50

(b) Rényi-Netzwerk – Stichprobengröße 250

(c) BM95 – Stichprobengröße 50

(d) BM95 – Stichprobengröße 250

Abbildung 5.9: Verteilungsdichte der VUS im Vergleich zwischen dem Rényi-Netzwerk und BM95. Im oberen Abschnitt befinden Daten für das Rényi-Netzwerk und in der unteren Reihe BM95. In der linken Spalte befinden sich die Daten für die Stichprobengröße 50 und in der rechten die für 250.

6 Diskussion

6.1 Relevanz der Vektorlänge

Betrachtet man die Ergebnisse der Abschnitte 5.2.1 bis 5.2.3 stellt sich ein zu erwartender Trend ein hinsichtlich der verwendeten Länge der Zeitreihen. Mit steigender Größe funktioniert die Rekonstruktion der Topologien besser. Ein deutlicher Sprung ist dabei in der Regel von 25 zu 50 und weiter zu Länge 250 feststellbar. Der Unterschied zwischen 250 und 500 ist jedoch in der Regel vernachlässigbar (daher wurde auch in Abschnitt 5.2.3 auf diese Vektorlänge verzichtet). Allerdings zeigt es auch das oft mit den kurzen Zeitreihen keine verlässlichen Aussagen getroffen werden können. Ein gutes Beispiel dafür ist die Transferentropie im Falle der Renyi-Netzwerke 5.6: Die Ergebnisse der VUS für eine Vektorlänge von 25 befinden sich nur knapp oberhalb von 0,5 und lassen damit keine Rekonstruktion zu. Allerdings verbessert sich der Wert schon bei einer Länge von 50 und ist bereits bei 250 nahezu eins, sodass hier eine vollständige Rekonstruktion des Graphen anhand der verschiedenen Zeitreihen möglich ist.

6.2 Einordnung der Informationsmaße

In diesem Abschnitt werden die Ergebnisse der einzelnen Informationsmaße und deren Fähigkeit zur Rekonstruktion der verschiedene Netzwerke beurteilt.

Die zeitverzögerten Korrelation ist ein naiver und naheliegender Ansatz, um die mögliche Abhängigkeit in zwei Zeitreihen zu erkennen. Der Vorteil der Methode ist, dass sie einfach zu implementieren und schnell zu berechnen ist. Allerdings gilt auch bei der zeitverzögerten Variante, dass non-lineare Abhängigkeiten nicht erkannt werden können.

Im Falle des einfachen DGL-Systems zeigt die Korrelation in keinem der getesteten Setups nennenswerten Rekonstruktionsergebnisse. Ebenso wie für die Barabási-Netzwerke erreicht das VUS praktisch keine höheren Werte als 0,5. Für die Rényi-Netzwerke stellt sich jedoch eine sehr verlässliche Rekonstruktion ein. Der VUS liegt hier selbst bei kurzen Vektoren im Schnitt über 0,9. Zumindest auch für kurze Zeitreihen der Rényi-Graphen lässt sich eine verlässliche Rekonstruktion festhalten. Bei längeren Vektoren ist auch ein zuverlässiges Ergebnis für das Netzwerk BM95 festzustellen. Im gesamten Vergleich schneidet die Korrelation tatsächlich am besten ab, da sie insgesamt die meisten VUS-Werte höher als 0,8 aufweist.

Für die Kreuzkorrelation ist das Netzwerk BM95 auch das einzige, das eine gute Rekonstruktion sowohl bei kleiner als auch großer Vektorlänge aufweist. Das Netzwerk BM160 wird ebenfalls in mittlerer Qualität rekonstruiert, alle anderen jedoch nicht. Zwar erreicht der VUS für das Rényi-Netzwerk auch Werte bis zu 0.75, aber nur in wenigen

Fällen und insgesamt ist der Wertebereich zu weit gestreut (vgl. 5.9b), um dem Maß selbst eine mittlere Rekonstruktionsqualität bei der Vektorlänge 250 zu attestieren.

Auch das letzte varianzbasierte Maß im Vergleich, die Informationsflussrate, kann nur das Netzwerk BM95 in sehr guter Qualität rekonstruieren. Selbst die Rényi-Netzwerke werden auch bei einer großen Vektorlänge nicht rekonstruiert. Siehe 6.2.1). Insgesamt schneidet damit die Informationsflussrate am schlechtesten von allen getesteten Maßen ab.

Das erste der beiden entropiebasierten Maße ist die TDMI. Das Rényi-Netzwerk wird in sehr guter Qualität rekonstruiert. Die Netzwerke BM12 und BM95 ergeben noch mittlere VUS-Werte bei einer Vektorlänge von 250.

Das zweite ist die Transferentropie. Von allen verwendeten Maßen ist es das Einzige, welches gute bis sehr gute Ergebnisse für die Rekonstruktion der Barabási-Graphen liefert. Interessanterweise ist dies auch das einzige Beispiel für eine erfolgreiche Rekonstruktion mittels Z-Score (Vektorlänge 250). Die Normierung der TE kann jedoch in keinem der getesteten Netzwerke einen VUS aufweisen, der eine erfolgreiche Rekonstruktion erlaubt. Die Werte für kurze Vektoren liegen im Falle von Rényi und Barabási nur knapp unter dem Grenzwert 0,8, sodass man hier durchaus auch von einem sehr guten VUS sprechen kann.

Insbesondere im DGL-System ergibt die TE im Vergleich mit den anderen Maßen ein konsistenteres Bild. Hier scheint das Ergebnis bei den anderen Maßen stark von den verwendeten Parametern für die Simulation abzuhängen. Allerdings ist der VUS dennoch nur im mittleren bis guten Bereich angesiedelt.

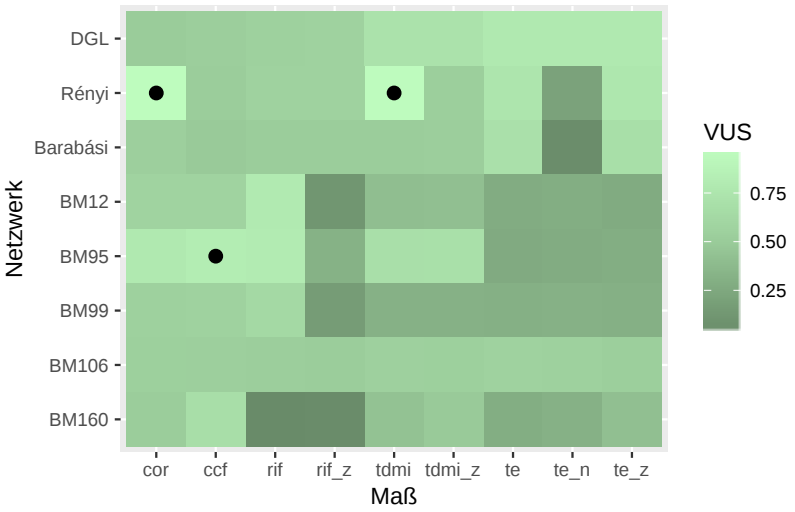
6.2.1 Normierung der Transferentropie

Die Normierung der Transferentropie zeigt in den Fällen des einfachen DGL-Systems und den BioModels-Netzwerken keinen Unterschied in der Leistungsfähigkeit zur Rekonstruktion und bewegt sich auf dem gleichen Niveau wie die Z-Score Ergebnisse und der reine Transferentropiewert. Im Falle der Markov-Netzwerke (Abschnitt 5.2.2) verschlechtert sich das Ergebnis deutlich. Eine mögliche Ursache könnte sein, dass die Optimierung zum Finden der maximalen und minimalen Transferentropien mehr Iterationen benötigt im Falle dieser Netzwerktypen.

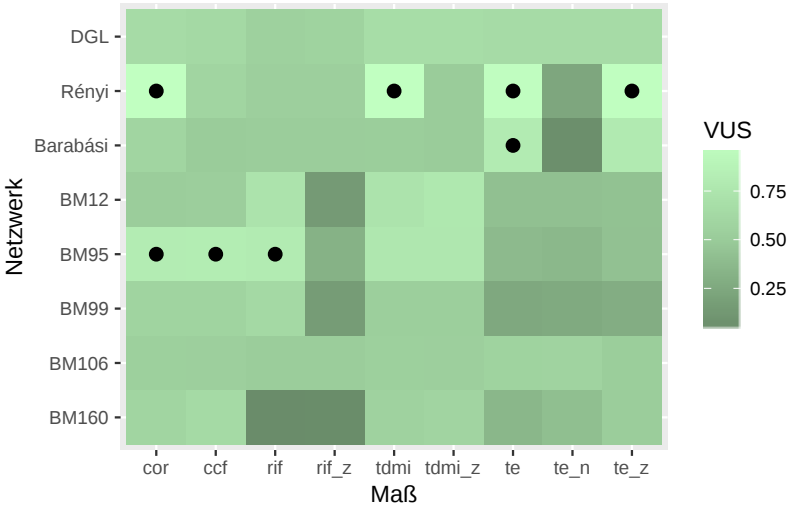
6.2.2 Einordnung Z-Scores

Benutzt man die Z-Score-Filterung anstelle der Maße selbst zur Rekonstruktion, lässt sich damit keine Rekonstruktion mehr durchführen. Es sind dabei alle drei Maße (RIF, TDMI und TE) betroffen sowie alle getesteten Netzwerke. Tatsächlich sorgte eine Grenze von einer 2σ-Umgebung dafür, dass die Rekonstruktion nicht möglich war. Das gleiche Ergebnis findet sich auch bei RIF und TDMI wieder.

Betrachtet man den Unterschied im VUS zwischen Transferentropie und Z-Score der Transferentropie (also keine Filterung), ist feststellbar, dass beide die gleiche Fähigkeit zur Rekonstruktion an den Tag legen, obwohl der Wert des Z-Scores häufig unter der



(a) Vektorlänge = 50



(b) Vektorlänge = 250

Abbildung 6.1: Zusammenfassung der VUS Werte aus Abschnitt 5.2. Die Heatmaps zeigen die Höhe des VUS an. Hohe Werte sind dabei hell und niedrige dunkel abgebildet. Zusätzlich wurden Werte die über 0,8 liegen mit einem Punkt gekennzeichnet. Abbildung 6.1a zeigt die Auswertung für eine Stichprobengröße von 50 und 6.1b für 250. Die gezeigten Maße sind: Kreuzkorrelation (ccf), Korrelationskoeffizient (cor), Informationsflussrate (rif), Z-Score der Informationsflussrate (rif_z), Zeitverzögerte Transinformation (tdmi), Z-Score der TDMI (tdmi_z), Transferentropie (te), normierte TE (te_n), Z-Score der TE (te_z). Auf der y-Achse sind die eingesetzten Netzwerke vermerkt. Das einfache DGL-System, die Markov-Ketten Netzwerke, sowie die Modelle aus der BioModels Database.

Signifikanzgrenze liegt. Für die TDMI trifft dies nur in einigen Fällen der BioModels-Database-Netzwerke zu. Der Z-Score der Informationsflussrate schneidet in jedem Fall schlechter ab.

Dieses Resultat ist insofern schwierig einzuordnen, als das der Z-Score ein erprobtes Mittel ist um die Signifikanz von Messergebnissen zu beurteilen. Hier konnte jedoch mit einem Nullmodell, welches auf randomisierten Adjazenmatrizen basiert, und der Volume-Under-Surface gezeigt werden, dass eine Rekonstruktion sehr gut möglich ist, trotz niedriger Z-Scores (siehe Abb. 5.6 und 5.7).

6.3 Fazit

Für die hier durchgeführte Analyse wurden zwei Erweiterungen für die Statistiksoftware `R` entwickelt. Diese dienen der Simulation von biologischen Netzwerken und der effizienten Berechnung der Transferentropie und einem Signifikanzmaß, dem Z-Score.

Bei der abschließenden Betrachtung der Analyse muss festgehalten werden, dass keines der verwendeten Maße für *alle* getesteten Netzwerke überzeugende Ergebnisse liefert. Die Transferentropie bietet Vorteile bei der Rekonstruktion der Markov-Systeme. Sie zeigt bei kurzen Zeitreihen des einfachen DGL-Systems, eine gute VUS und damit noch das beste Ergebnis. Daher wäre ein ähnliches Verhalten bei den biologischen Netzen zu erwarten gewesen. Das Gegenteil ist jedoch der Fall. Die zeitverzögerte Korrelation scheint bei längeren Vektoren die verlässlichste Methode zu sein.

Eine Ursache für das inkonsistente Abschneiden der verschiedene Maße könnten Netzwerkmotive sein, die je nach eingesetzter Methode unterschiedliche Auswirkungen haben. Beispielsweise enthalten die Graphen aus der BioModels Database in der Regel viele Zyklen (Feedback-Loops), während die auf Markov-Ketten basierten Netzwerke selten Kreise enthalten. Gegen diese Theorie spricht allerdings, dass das einfache DGL-System letztlich nur zwei gekoppelte Kreise darstellt und dies von der Transferentropie in mittlerer bis guter Qualität rekonstruiert werden kann. Ein Problem bei der Einordnung dieses Beispielsystems könnte hier jedoch die insgesamt geringe Größe des Netzwerks sein, da die VUS dort aufgrund der kleinen Matrix schnell eine niedrigen Wert ergibt, sobald bereits ein Knoten falsch erkannt wird. Im Falle des Barabási-Modells gibt es wenige Knoten mit hoher Zentralität. Dies kann ebenfalls dazu führen, dass die Informationsflüsse zu diesen Knoten einander überlagern und dadurch schlechter erkannt werden.

7 Ausblick

7.1 Ansätze zur Verbesserung der Rekonstruktionsleistung

Mögliche Ansätze um die Rekonstruktionsleistung zu verbessern, sollen in diesem Abschnitt angeführt werden. Eine einfache, aber in realen Applikationen oft nicht machbare Verbesserung, wäre der offensichtliche Ansatz vergrößerte Stichproben zu verwenden. Um diesen Problem des sogenannten *finite size effects* zu begegnen wurden bereits verschiedene Lösungsansätze in der Literatur vorgeschlagen, sind jedoch in der Regel spezifisch zu dem betrachteten Problem. In [71] schlagen die Autoren vor, aus kurzen Intervallen längere Trajektorien zu erzeugen. Diese Methode baut auf den von Eckmann et al. [34] vorgestellten Rekurrenzplots auf. Dies sind jedoch nur indirekte Ansätze um das Problem der begrenzten Datenpunkte anzugehen. Eine Methode, die auf kurzen Datensätzen reproduzierbare und verlässliche Ergebnisse liefert, wäre daher wünschenswert.

Die Transferentropie zeigt in bestimmten Szenarien (siehe Graphik 6.1a) bereits vielversprechende Ansätze. Zwar liefert sie dort teils gute bis sehr gute Ergebnisse bei der Rekonstruktion, allerdings ist die Effektivität auf bestimmte Netzwerktypen beschränkt. Hier bedarf es weiterer Untersuchungen, um die 6.3 aufgestellten Thesen (z.B. Abhängigkeit von bestimmten Eigenschaften der Netzwerke wie z. B. Zentralität oder Zyklen) zu verifizieren.

In [117] stellen die Autoren eine Variante der Transinformation vor, die drei Verteilungen in die Berechnung einbezieht anstatt zwei. Diese Erweiterung wäre auch für die TDMI und TE denkbar. Auch eine Erhöhung auf mehr als drei Verteilung kommt in Frage. Damit könnten Einflüsse auf Knoten festgestellt werden, die mehr als einen Ursprungsknoten haben.

Ein kombinierter Einsatz von zwei verschiedenen Maßen könnte ebenfalls ein verbessertes Rekonstruktionsergebnis zur Folge haben. Ein gestaffelter Ansatz bei dem z.B. zuerst für ein Knotenpaar das beste Ergebnis der Kreuzkorrelationsfunktion gefunden wird und anhand dessen eine Schrittweite für eine darauf folgende Transferentropieberechnung festgelegt wird. Damit würde der Tatsache Rechnung getragen, dass nicht alle Interaktionen in einem biologischen Netzwerk zu gleichen Zeitintervallen passieren.

Ein nicht zu unterschätzender Faktor beim Anfertigen dieser Arbeit waren Schwierigkeiten mit den Ausgangsdaten. Selbst die kurierten Netzwerke der BioModels Database waren mit unter nicht in das von uns verwendete C-Format konvertierbar. Die Problematik dabei ist entweder, dass es nicht immer strikte Regeln gibt anhand derer die biologischen Systeme einheitlich beschrieben werden oder fehlende Erfahrung im Umgang mit XML Formaten, da häufig Naturwissenschaftler und nicht Informatiker an den

Modellen arbeiten. In ihrer Studie über molekulare Reaktionsnetzwerke konnten Kaleta et al. [66] in 22% der Systeme der BioModels Database (Stand 2009) Inkonsistenzen feststellen, die zu deutlichen Fehlern bei der Simulation führen können. Keines der in dieser Arbeit verwendeten Netzwerken wird jedoch von den Autoren erwähnt.

Ob diese Inkonsistenzen womöglich zu einem weiteren Fehler, der bei dem Export aus COPASI aufgetreten ist führen, ist fraglich: Die Differentialgleichungen wurden manchmal mehrfach in die C-Datei übertragen. Es konnte keine spezielle Ursache in den Ursprungsdaten ausgemacht werden, jedoch stellte sich das Problem als spezifisch für bestimmte Dateien dar, sodass dies vermutlich auf einen Fehler in der Exportfunktion von COPASI zurückzuführen ist. Dies bedeutete das ein weiterer manueller Schritt notwendig war, um die Daten vor der Auswertung zu kurieren.

Ein wichtiger Schritt ist hierbei der konsequente Aufbau und Erweiterung von Ontologien in den Datenbanken. Dadurch können die Daten vielfältig in unterschiedlichem Kontext genutzt werden. Dies ist in Übereinstimmung mit den Grundgedanken der Systembiologie, da es u.a. erlaubt in Simulationen auf Mikro- und Makro-Ebene zu verknüpfen [57]. Die Anreicherung mit diesen Metainformationen kann wiederum nützlich sein für Algorithmen zur Auswertung der Daten. Da es derzeit kein Standardformat für biologische Systeme gibt (z.B. SBML vs. CellML [78]), sind Wissenschaftler häufig darauf angewiesen zwischen verschiedenen Formaten zu konvertieren um vergleichbare Ergebnisse zu erzeugen oder bei der Auswahl der Software Kompromisse einzugehen.

7.1.1 Parallelisierung mittels GPU

Aufbauend auf den Erkenntnissen aus der Entwicklung der C++ Bibliothek in Abschnitt 4.2 resultierte eine Studienarbeit, die ebenfalls von mir betreut wurde. Sie beschäftigte sich mit der Möglichkeit der Parallelisierung mittels NVIDIA's Cuda Technologie [88]. Eine Geschwindigkeitssteigerung für die Berechnung des Z-Scores für entropiebasierte Maße durch GPUs (Graphic Processing Units) konnte bereits von Wächter et. al. gezeigt werden [116, 117]. Desweiteren gibt es bereits andere R Pakete die einen Nutzen, vor allem im Bereich der linearen Algebra, aus GPU-beschleunigter Berechnung ziehen [32, 100]. Das R Paket GPUTOOLS bietet unter anderem bereits ein Test zur Granger-Kausalität und Korrelationskoeffizienten an, aber noch keine Entropie basierten Maße [21]. Als nächster Schritt wurde daher in dieser Arbeit das Augenmerk auf die Normierung der Transferentropie gelegt. Der Vorteil von GPUs ist die hochparallele Struktur des Prozessors. Dies bietet sich für den Prozess des in Abschnitt 4.3.1 vorgestellten Greedy-Optimierers an, da die einzelnen Schritte datenunabhängig voneinander durchgeführt werden können. Allerdings gibt es Einschränkungen auf Seiten des verfügbaren Speichers pro Prozessoreinheit. Beides zusammen macht eine aufwendige Anpassung des Codes notwendig. Ein von mir betreutes Praktikum konnte bereits vielversprechende Ansätze zeigen, sodass eine zukünftige Erweiterung des TransferEntropyPT Pakets damit in Frage kommt.

8 Anhang

8.1 Datensätze Markov-Netzwerke

Datensatz	p^+	p^0	Datensatz	p^+	p^0
1	0.46	0.15	14	0.53	0.29
2	0.52	0.01	15	0.76	0.16
3	0.64	0.15	16	0.60	0.33
4	0.84	0.35	17	0.73	0.26
5	0.42	0.14	18	0.90	0.31
6	0.84	0.19	19	0.53	0.22
7	0.87	0.24	20	0.77	0.21
8	0.70	0.20	21	0.86	0.32
9	0.68	0.07	22	0.43	0.01
10	0.34	0.33	23	0.69	0.19
11	0.42	0.27	24	0.38	0.29
12	0.41	0.32	25	0.46	0.28
13	0.71	0.04			

Tabelle 8.1: Die Tabelle enthält die Übergangswahrscheinlichkeiten der 25 zufällig generierte Datensätze für Markov-Netzwerke. Für eine bessere Lesbarkeit ist die Wahrscheinlichkeit p^+ die Übergangswahrscheinlichkeit für Knoten mit mindestens einer Ausgangskante $D^+(v) > 0$. Für Knoten ohne Ausgangskante $D^+(v) = 0$ sind die Wahrscheinlichkeiten in Spalte p^0 notiert.

8.2 Verwendete biologische Netzwerke

Die Grafiken wurden mit der Software Cytoscape erzeugt. Die Pfeiltypen entstammen der Annotation der SBML Daten. Wie in Abschnitt 3.5 beschrieben wurden die Grafiken für die Berechnung in vereinfachte Adjazenzmatrizen überführt.

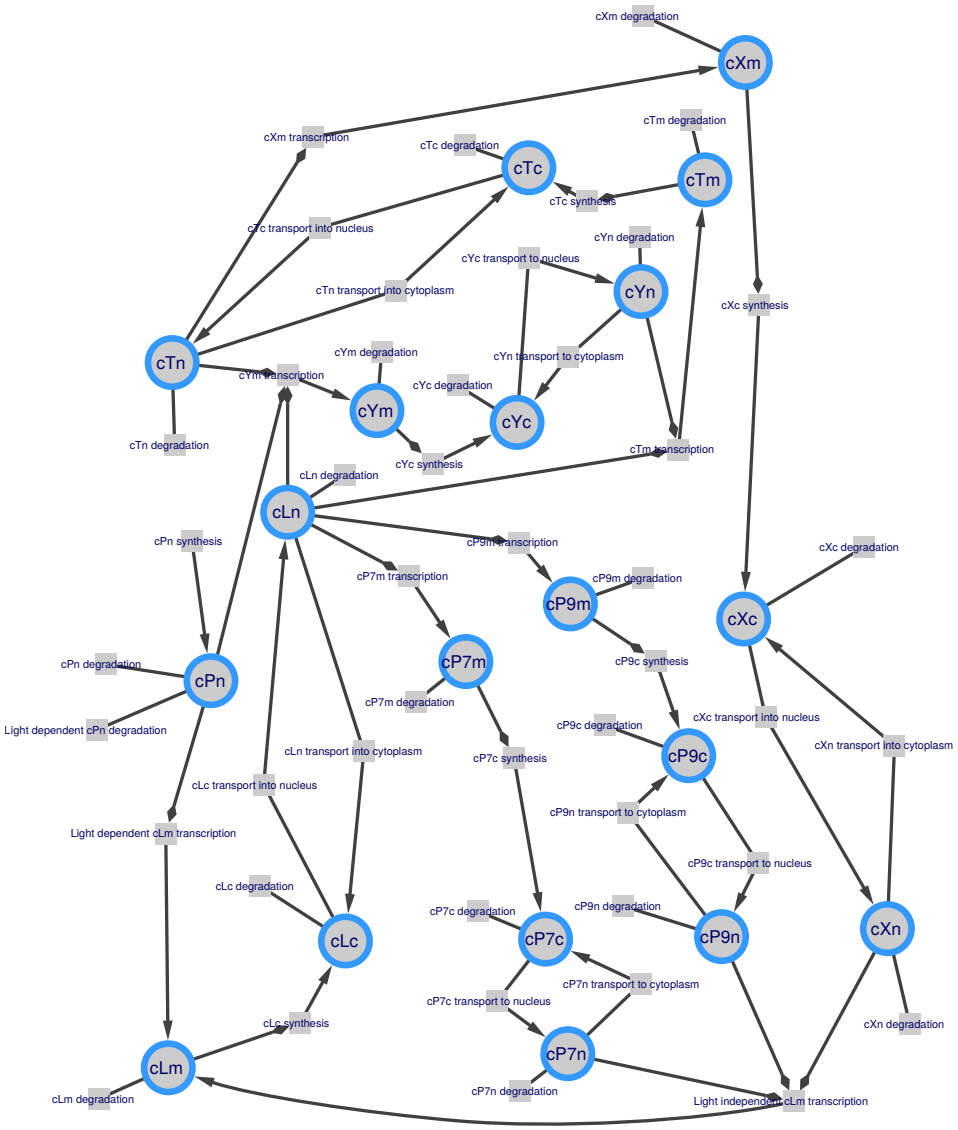


Abbildung 8.1: Graph des Netzwerks BM95

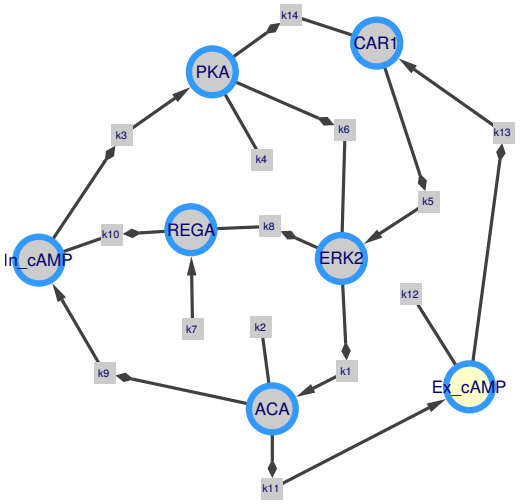


Abbildung 8.2: Graph des Netzwerks BM99

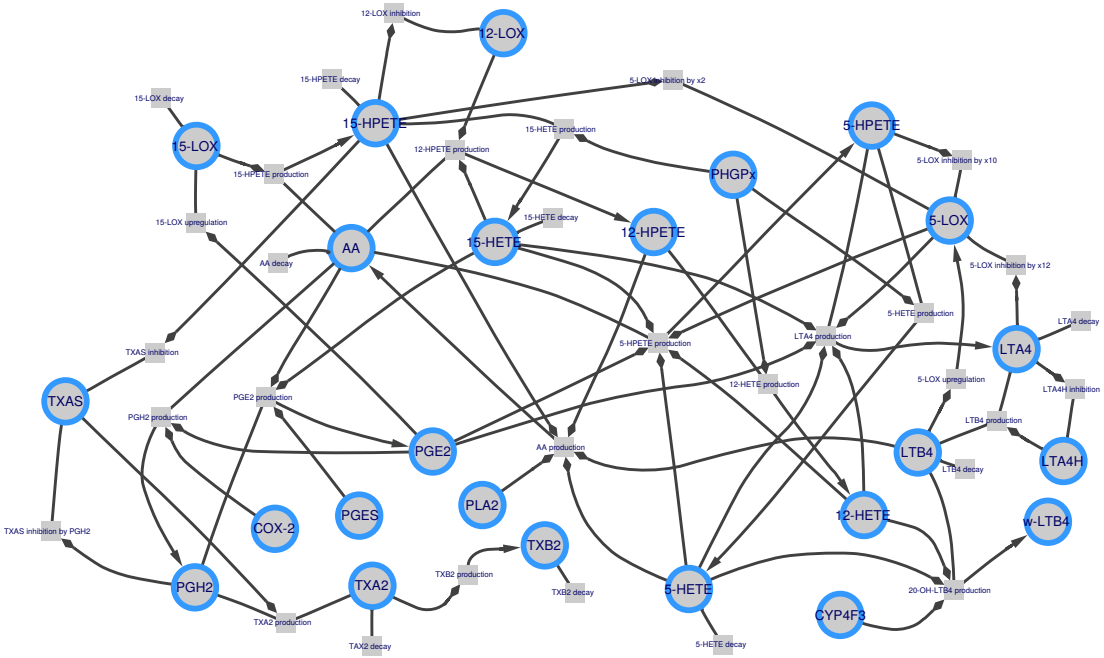


Abbildung 8.3: Graph des Netzwerks BM106

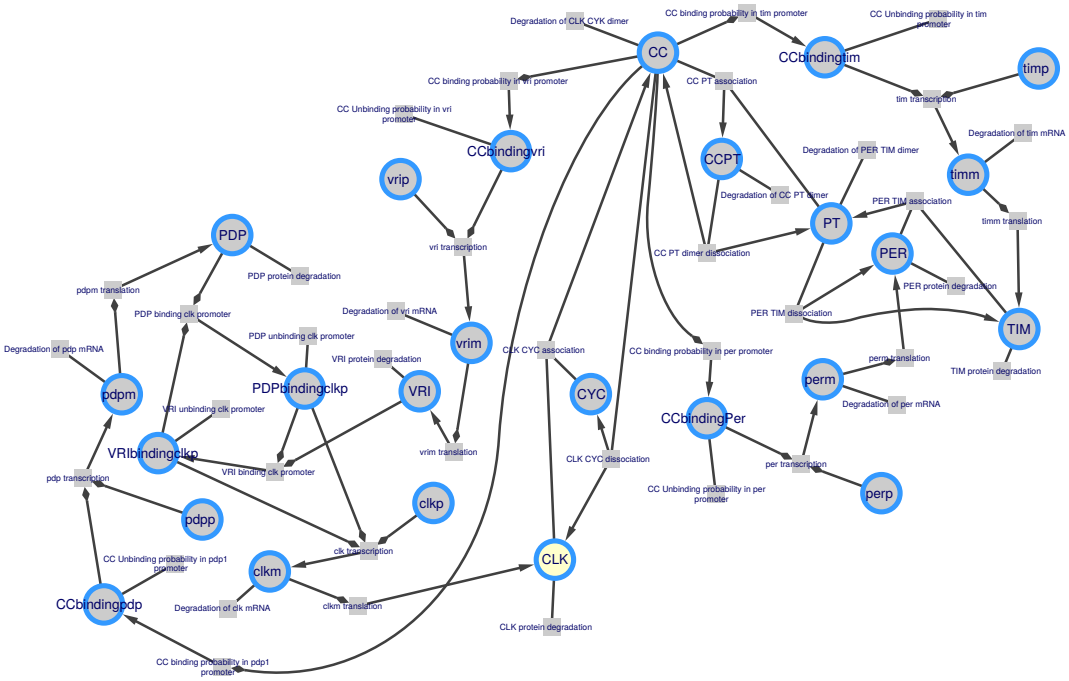


Abbildung 8.4: Graph des Netzwerks BM160

8.2.1 Code-Beispiele

Auflistung 8.1: Beispielcode TransferEntropyPT

```
## Erzeugen der Coupled Logistic Map
c.map <- coupled.map(n = 500, xi = 0.7, alpha = 0.8)
plot(c.map)

## Binning der Rohdaten.
p.map <- partition.data(c.map, bins = 8)
plot(p.map)

## Berechnung der Transferentropie mit Z-Score und Percentile.
te.map <- get.te(c.map, bins = 8, wsize = 1, zscore = 500, perc = T)

>>te.map

$te
       1->      2->
->1 0.0000000 0.8900929
->2 0.2182951 0.0000000

$mean
       1->      2->
->1 0.0000000 0.5959438
->2 0.6081191 0.0000000

$sd
       1->       2->
->1 0.00000000 0.03019493
->2 0.03094065 0.00000000

$zscore
       1->    2->
->1   0.00000 9.74167
->2 -12.59909 0.00000

$percentile
    1->  2->
->1   0 0.998
->2   0 0.000
```

Auflistung 8.2: Code Beispiel des C-Controllers, der die Pointerübergabe zwischen Fortran Solver und dem Shared Object des Netzwerkmodells übernimmt. `odefun` ist die zu integrierende Funktion und `setFA` verändert einen globalen Zeiger. Der Aufbau erlaubt dem Benutzer außerdem über den Parameter `beta` eine eigene Rauschfunktion dem Solver zu übergeben. Diese wird ebenfalls als Objekt in die gemeinsame Bibliothek geladen und kompiliert (Code Stefan Gries in [16]).

```
...
// forward-define Fortran functions
void setFA(void (**)(double*, double**, double**, double*));
void setDrift(void (**)(double*, double**, double*));
void setBeta(double*);
...
// passing parameters as s-expressions
SEXP launchSim(SEXP infile, ..., SEXP trmethod)
{
...
    void * clib;
    PROTECT(infile = coerceVector(infile, STRSXP));
    // load given shared object and extract needed function symbol
    clib = dlopen(CHAR(STRING_ELT(infile, 0)), RTLD_NOW | TLD_LOCAL);
    // load dynamic library "infile"
    PROTECT(fnname = coerceVector(fnname, STRSXP));
    // allocation of 'ode_fun' using "fnname" function from "infile"
    *(void **)(&ode_fun) = dlsym(clib, CHAR(STRING_ELT(fnname, 0)));
       UNPROTECT(2);
    // pass function pointer to fortran
    setFA(&ode_fun);
    void * ex_drift;
    if(isNumeric(beta)) {
        PROTECT(beta = coerceVector(beta, REALSXP));
        // set beta value for internal noise
        setBeta(REAL(beta));
        UNPROTECT(1);
    } else if(isString(beta)) {
        // basically do the same with drift function as with ODE
        // function, if custom one is to be used.
        // function pointer declaration 'drift_fun':
        void (*drift_fn)(double*, double**, double*);
        PROTECT(beta = coerceVector(beta, STRSXP));
        PROTECT(betafname = coerceVector(betafname, STRSXP));
        // load dynamic library "beta"
        ex_drift = dlopen(CHAR(STRING_ELT(beta, 0)), RTLD_NOW |
           RTLD_LOCAL);
        // load "betafname" symbol from "beta" into 'drift_fun'
        *(void**)(&drift_fn) = dlsym(ex_drift,
           CHAR(STRING_ELT(betafname, 0)));
        UNPROTECT(2);
        // set drift function pointer in Fortran
        setDrift(&drift_fn);
    } else {fprintf(stderr,"Error in beta argument"); exit(1); }
...
}
```

Glossar

ACF Autocorrelation Function. 18

AUC Area under Curve. 25

BMD BioModels Database. 2, 10, 14, 38

CCF Cross Correlation Function. 18

COPASI Complex Pathway Simulator. 45–47, 86

COPD Chronisch obstruktive Lungenerkrankung. 13

DGL Gewöhnliche Differentialgleichung. 27, 71, 72

DNA Deoxyribonucleic acid. 1, 3, 9

EMBL-EBI European Bioinformatics Institute. 10, 38

FBA Flux-Balance Analysis. 45

FN False Negative. 25

FP False Positive. 25

FPR False Positive Rate. 25

GCC GNU Compiler Collection. 48

GPU graphics processing unit. 51

HMM Hidden Markov Model. 32, 33, 35

HPC High Performance Computing. 10

MI Mutual Information (Transinformation). 21, 50

mRNA messenger ribonucleic acid. 13

NCBI National Center of Biotechnology. 10

ODE Ordinary Differential Equation. 27

RIF Rate of Information Flow (Informationsflussrate). 72

RK Runge-Kutta. 30

RNA ribonucleic acid. 1, 3

ROC Receiver-Operating-Characteristic. 25, 26, 85

SBML Systems Biology Markup Language. 10, 14, 45, 46

SDE Stochastic Differential Equation. 29

SIMD Single Instruction, Multiple Data. 48

SOMNIBIEN Simulation Of Metabolic Networks Influenced By Internal And External Noise. 7, 14, 45–49, 86

TDMI Time Delayed Mutual Information (zeitverzögerte Transinformation). 20–22, 72

TE Transferentropie. 14, 22

TN True Negative. 25

TP True Positive. 25

TPR True Positive Rate. 25

VUS Volumen under Surface. 25, 26, 71, 72

WGC Wiener-Granger Causality. 12

XML Extensible Markup Language. 10, 14

Abbildungsverzeichnis

Literaturverzeichnis

[1] Jens Ackermann, Jens Ackermann, Paul Baecher, Thorsten Franzel, Michael Goesele, and Kay Hamacher. Massively-Parallel Simulation of Biochemical Systems. *Proceedings of Massively Parallel Computational Biology on GPUs. Lecture Notes in Informatics (LNI), Lübeck, Germany*, 154:739–750, 2009.

[2] Alfred J. Lotka. Elements of Physical Biology. *Williams and Wilkins Company*, page 435, 1925.

[3] G.M. Amdahl. Validity of the single-processor approach to achieving large scale computing capabilities. *AFIPS Conference Proceedings*, 30:483–485, 1967.

[4] Sophie Arnaud-Haond, Yann Moalic, Christian Barnabé, Francisco José Ayala, and Michel Tibayrenc. Discriminating micropathogen lineages and their reticulate evolution through graph theory-based network analysis: The case of Trypanosoma cruzi, the agent of Chagas Disease. *PLoS ONE*, 9(8), 2014.

[5] Marcel Ausloos and Michel Dirickx. *The logistic map and the route to chaos: from the beginnings to modern applications*. Springer Science and Business Media, 2006.

[6] Seung Ki Baek, Woo-Sung Jung, Okyu Kwon, and Hie-Tae Moon. Transfer Entropy Analysis of the Stock Market. *ArXiv Physics e-prints*, sep 2005.

[7] A. L. Barabási, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications*, 311(3-4):590–614, 2002.

[8] Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(October):509–512, 1999.

[9] Lionel Barnett, Adam B. Barrett, and Anil K. Seth. Granger causality and transfer entropy Are equivalent for gaussian variables. *Physical Review Letters*, 103(23):2–5, oct 2009.

[10] Alex Bavelas. Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America*, 22(6):725–730, 1950.

[11] Dennis A. Benson, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. GenBank. *Nucleic Acids Research*, 45(D1):D37–D42, 2017.

[12] H M Berman, J Westbrook, Z Feng, G Gilliland, T N Bhat, H Weissig, I N Shindyalov, and P E Bourne. The Protein Data Bank. *Structural Bioinformatics*, 28(1):181–198, 2005.

[13] Anastasios Bezerianos, Andrei Dragomir, and Panos Balomenos. *Computational Methods for Processing and Analysis of Biological Pathways*. Number May. Springer, Cham, 2017.

[14] P Boba, P Weil, F Hoffgaard, and K Hamacher. Co-Evolution in HIV Enzymes. *Proc. of BIOINFORMATICS 2010*, pages 39–47, 2010.

[15] Patrick Boba, Dominik Bollmann, Daniel Schoepe, Nora Wester, Jan Wiesel, and Kay Hamacher. Efficient computation and statistical assessment of transfer entropy. *Frontiers in Physics*, 3(March):1–9, 2015.

[16] Patrick Boba, Stefan Gries, and Kay Hamacher. R as an Integration Tool in High Performance Computing – Lessons Learned. In *Lecture Notes in Informatics (LNI)*, pages 137–148, Stuttgart, 2014. Bonner Köllen Verlag.

[17] Patrick Boba and Kay Hamacher. TransferEntropyPT: An R package to assess transfer entropies via permutation tests. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10545 LNBI, pages 285–290. Springer, Cham, sep 2017.

[18] Patrick Boba, Philipp Weil, Franziska Hoffgaard, and Kay Hamacher. Intra- and inter-molecular coevolution: The case of HIV1 protease and reverse transcriptase. *Communications in Computer and Information Science*, 127 CCIS:356–366, 2011.

[19] Sebastian Bremm, Tatiana von Landesberger, Martin Heß, Tobias Schreck, Philipp Weil, and Kay Hamacher. Interactive Visual Comparison of Multiple Trees. In *IEEE Visual Analytics Science and Technology*, pages 31–40, 2011.

[20] Steven L. Bressler and Anil K. Seth. Wiener-Granger Causality: A well established methodology. *NeuroImage*, 58(2):323–329, sep 2011.

[21] Joshua Buckner, Justin Wilson, Mark Seligman, Brian Athey, Stanley Watson, and Fan Meng. The gputools package enables GPU computing in R. *Bioinformatics*, 26(1):134–135, 2009.

[22] Kevin Burrage and John C Butcher. Stability criteria for implicit Runge–Kutta methods. *SIAM Journal on Numerical Analysis*, 16(1):46–57, 1979.

[23] John C Butcher. On Runge-Kutta processes of high order. *Journal of the Australian Mathematical Society*, 4(02):179–194, 1964.

[24] Carter T. Butts. Revisiting the foundations of network analysis. *Science*, 325(5939):414–416, 2009.

[25] Han-Yu Chuang, Matan Hofree, and Trey Ideker. A Decade of Systems Biology. *Annual Review of Cell and Developmental Biology*, 26(1):721–744, nov 2010.

[26] William S. Cleveland and E. Grosse. Computational methods for local regression. *Statistics and Computing*, 1(1):47–62, 1991.

[27] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Sy:1695, 2006.

[28] Peter Csermely, Tamás Korcsmáros, Huba J.M. Kiss, Gábor London, and Ruth Nussinov. Structure and dynamics of molecular networks: A novel paradigm of drug discovery: A comprehensive review. *Pharmacology and Therapeutics*, 138(3):333–408, 2013.

[29] Kathryn D. Curtin, Zuoshi J. Huang, and Michael Rosbash. Temporally regulated nuclear entry of the Drosophila period protein contributes to the circadian clock. *Neuron*, 14(2):365–372, 1995.

[30] L. Dagum and R. Menon. OpenMP: an industry standard API for shared-memory programming. *IEEE Computational Science and Engineering*, 5(1):46–55, 1998.

[31] Matthias Dehmer, Frank Emmert-Streib, Armin Graber, and Armindo Salvador. *Applied Statistics for Network Biology: Methods in Systems Biology*, volume 1. Wiley-Blackwell, Weinheim, 2011.

[32] Charles Determan Jr. gpuR: GPU Functions for R Objects, 2017.

[33] P. Dwight Kuo, Wolfgang Banzhaf, and André Leier. Network topology and the evolution of dynamics in an artificial genetic regulatory network model created by whole genome duplication and divergence. *BioSystems*, 85(3):177–200, 2006.

[34] J. P. Eckmann, O. Oliffson Kamphorst, and D. Ruelle. Recurrence plots of dynamical systems. *Epl*, 4(9):973–977, 1987.

[35] Dirk Eddelbuettel and Romain François. Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, 40(8):1–18, apr 2011.

[36] Michael B Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, jan 2000.

[37] F Emmert-Streib and M Dehmer. *Information Theory and Statistical Learning*. Springer, 2008.

[38] Paul Erdös and Alfréd Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[39] Hucka Et Al. and Hucka Et Al. The system biology markup language (SBML): A medium for represenation and exchange of biochmical network models. *Bioinf*, 19(4):524–531, 2003.

[40] A Finney and M Hucka. Systems biology markup language: Level 2 and beyond. *Biochemical Society Transactions*, 31(Pt 6):1472–3, 2003.

[41] Andrew M. Fraser and Harry L. Swinney. Independent coordinates for strange attractors from mutual information. *Physical Review A*, 33(2):1134–1140, 1986.

[42] L.C Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, 1977.

[43] Akira Funahashi, Mineo Morohashi, Hiroaki Kitano, and Naoki Tanimura. CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *Biosilico*, 1(5):159–162, 2003.

[44] Eva Gehrmann, Christine Gläßer, Yaochu Jin, Bernhard Sendhoff, Barbara Drossel, and Kay Hamacher. Robustness of glycolysis in yeast to internal and external noise. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 84(2):21913, aug 2011.

[45] Tamar Geiger, Juergen Cox, and Matthias Mann. Proteomics on an Orbitrap Benchtop Mass Spectrometer Using All-ion Fragmentation. *Molecular & Cellular Proteomics*, 9(10):2252–2261, 2010.

[46] I. M. Gelfand and A. M. Yaglom. On the General Definition of the Quantity of Information. In A N Shiryayev, editor, *Dokl. Akad. Nauk SSSR*, volume 111 of *Mathematics and Its Applications*, pages 2–5. Springer Netherlands, 1993.

[47] C W Granger. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica*, 37(3):424–438, 1969.

[48] Stefan Gries. *Simulation of noisy metabolic networks in R.* Bachelor thesis, TU Darmstadt, 2012.

[49] Olle Häggström. *Finite Markov chains and algorithmic applications*, volume 52. Cambridge University Press, 2002.

[50] Daniel W. Hahs and Shawn D. Pethel. Distinguishing anticipation from causality: Anticipatory bias in the estimation of information flow. *Physical Review Letters*, 107(12):128701, sep 2011.

[51] K. Hamacher. Information theoretical measures to analyze trajectories in rational molecular design. *Journal of Computational Chemistry*, 28(16):2576–2580, 2007.

[52] Kay Hamacher. Relating sequence evolution of HIV1-protease to its underlying molecular mechanics. *Gene*, 422(1-2):30–36, 2008.

[53] Kay Hamacher. A New Hybrid Metaheuristic – Combining Stochastic Tunneling and Energy Landscape Paving. In María J Blesa, Christian Blum, Paola Festa, Andrea Roli, and Michael Sampels, editors, *Hybrid Metaheuristics: 8th International*

Workshop, HM 2013, Ischia, Italy, May 23-25, 2013. Proceedings, pages 107–117. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[54] Paul E. Hardin, Jeffrey C. Hall, and Michael Rosbash. Feedback of the Drosophila period gene product on circadian cycling of its messenger RNA levels. *Nature*, 343(6258):536–540, 1990.

[55] Jürgen Hedderich and Lothar Sachs. *Angewandte Statistik: Methodensammlung mit R (German Edition)*. Springer-Verlag, 2011.

[56] Katerina Hlaváčková-Schindler, Milan Paluš, Martin Vejmelka, and Joydeep Bhattacharya. Causality detection based on information-theoretic approaches in time series analysis. *Physics Reports*, 441(1):1–46, 2007.

[57] Robert Hoehndorf, Michel Dumontier, John H Gennari, Sarala Wimalaratne, Bernard De Bono, Daniel L Cook, and George Gkoutos. Formalizing Systems Biology Models with Biomedical Ontologies. *Systems Biology*, 5(124), 2011.

[58] Franziska Hoffgaard, Philipp Weil, and Kay Hamacher. BioPhysConnectoR: Connecting Sequence Information and Biophysical Models. *BMC Bioinformatics*, 11(1):199, 2010.

[59] M Hollander and D A Wolfe. *Nonparametric Statistical Methods BT - Nonparametric Statistical Methods*, volume 751. John Wiley & Sons, 1973.

[60] Raquell Holmes. *A Cell Biologist Guide to Modeling & Bioinformatics*. Wiley-Interscience, Hoboken, N.J, 2007.

[61] Stefan Hoops, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, and Ursula Kummer. COPASI - A COmplex PAthway SImulator. *Bioinformatics*, 22(24):3067–3074, 2006.

[62] Daniel Horn. A correction for the effect of tied ranks on the value of the rank difference correlation coefficient. *Journal of Educational Psychology*, 33(9):686, 1942.

[63] Shinya Ito, Michael E. Hansen, Randy Heiland, Andrew Lumsdaine, Alan M. Litke, and John M. Beggs. Extending transfer entropy improves identification of effective connectivity in a spiking cortical network model. *PLoS ONE*, 6(11):e27431, jan 2011.

[64] Hawoong Jeong, B Tombor, Reka Albert, Zoltan N Oltvai, and Albert-Laszla Barabasi. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[65] Jaeseung Jeong, John C. Gore, and Bradley S. Peterson. Mutual information analysis of the EEG in patients with Alzheimer's disease. *Clinical Neurophysiology*, 112(5):827–835, may 2001.

[66] Christoph Kaleta, Stephan Richter, and Peter Dittrich. Using chemical organization theory for model checking. *Bioinformatics*, 25(15):1915–1922, 2009.

[67] Ross Kindermann and J. Laurie Snell. Markov Random Fields and Their Applications. *American mathematical society*, 1980.

[68] Justin B Kinney, Gasper Tkacik, and Curtis G Callan. Precise physical models of protein-DNA interaction from high-throughput data. *Proceedings of the National Academy of Sciences of the United States of America*, 104(2):501–506, 2007.

[69] Scott Kirkpatrick. Optimization by Simulated Annealing Optimization by Simulated Annealing. *JSTOR*, 220(January 1983):671–680, 2014.

[70] Richard Kleeman. Information flow in ensemble weather predictions. *Bulletin of the American Meteorological Society*, 88(4):491–492, mar 2007.

[71] C. Komalapriya, M. Thiel, M. C. Romano, N. Marwan, U. Schwarz, and J. Kurths. Reconstruction of a system's dynamics from short trajectories. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 78(6):1–11, 2008.

[72] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[73] M. T. Laub and W. F. Loomis. A Molecular Network That Produces Spontaneous Oscillations in Excitable Cells of Dictyostelium. *Molecular Biology of the Cell*, 9(12):3521–3532, 1998.

[74] M. Lawrence. Quick Introduction to the rsbml Package, 2007.

[75] Chen Li, Marco Donizelli, Nicolas Rodriguez, Harish Dharuri, Lukas Endler, Vijayalakshmi Chelliah, Lu Li, Enuo He, Arnaud Henry, Melanie I. Stefan, Jacky L. Snoep, Michael Hucka, Nicolas Le Novère, and Camille Laibe. BioModels Database: An enhanced, curated and annotated resource for published quantitative kinetic models. *BMC Systems Biology*, 4(1):92, jan 2010.

[76] F. Li, T. Long, Y. Lu, Q. Ouyang, and C. Tang. The yeast cell-cycle network is robustly designed. *Proceedings of the National Academy of Sciences*, 101(14):4781–4786, apr 2004.

[77] X. San Liang. Unraveling the cause-effect relation between time series. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 90(5):052150, 2014.

[78] Catherine M. Lloyd, James R. Lawson, Peter J. Hunter, and Poul F. Nielsen. The CellML Model Repository. *Bioinformatics*, 24(18):2122–2123, 2008.

[79] Carlos F Lopez, Jeremy L Muhlich, John A Bachman, and Peter K Sorger. Programming biological models in Python using PySB. *Molecular systems biology*, 9(646):646, feb 2013.

[80] D. A. Lury and R. A. Fisher. Statistical Methods for Research Workers. *The Statistician*, 21(3):229, 2006.

[81] Donatello Materassi. Norbert Wiener's legacy in the study and inference of causation. *2014 IEEE Conference on Norbert Wiener in the 21st Century*, pages 1–6, 2014.

[82] Christopher L. McClendon, Lan Hua, Gabriela Barreiro, and Matthew P. Jacobson. Comparing conformational ensembles Using the Kullback-Leibler divergence expansion. *Journal of Chemical Theory and Computation*, 8(6):2115–2126, jul 2012.

[83] José L. Medina-Franco, Marc A. Giulianotti, Gregory S. Welmaker, and Richard A. Houghten. Shifting from the single to the multitarget paradigm in drug discovery. *Drug Discovery Today*, 18(9-10):495–501, 2013.

[84] Martin Meier-Schellersheim, Iain D. C. Fraser, and Frederick Klauschen. Multiscale modelling for biologists. *WIREs System Biology and Medicine*, 1(December):4–14, 2012.

[85] Andrew V Metcalfe and Paul S P Cowpertwait. *Introductory time series with R*. Springer, 2009.

[86] Gordon E. Moore. Progress in digital integrated electronics. *IEEE Solid-State Circuits Society Newsletter*, 11(3):36–37, 2009.

[87] S P Norsett. Aspects of parallel Runge-Kutta methods. Numerical Methods for Ordinary Differential Equations. In *Proceedings Aquila Symposium, Lecture Notes in Mathematics*, volume 1386. Springer-Verlag, 1989.

[88] Nvidia Corporation. NVIDIA CUDA C Programming Guide Version 3.2, 2010.

[89] Aisling O'Driscoll, Jurate Daugelaite, and Roy D. Sleator. 'Big data', Hadoop and cloud computing in genomics. *Journal of Biomedical Informatics*, 46(5):774–781, 2013.

[90] Lothar Papula. *Mathematik für Ingenieure und Naturwissenschaftler Band 2*. Vieweg+Teubner Verlag, 2011.

[91] Karl Pearson. Note on Regression and Inheritance in the Case of Two Parents. *Proceedings of the Royal Society of London (1854-1905)*, 58(347-352):240–242, 2006.

[92] Jonathan D. Phillips, Wolfgang Schwanghart, and Tobias Heckmann. Graph theory in the geosciences. *Earth-Science Reviews*, 143:147–160, 2015.

[93] Iva Pritišanac, Matteo T. Degiacomi, T. Reid Alderson, Marta G. Carneiro, Eiso Ab, Gregg Siegal, and Andrew J. Baldwin. Automatic Assignment of Methyl-NMR Spectra of Supramolecular Machines Using Graph Theory. *Journal of the American Chemical Society*, 139(28):9523–9533, 2017.

[94] Ale Prokop and Bela Csukas. *Editorial and introduction*, volume 1. WILEY-VCH Verlag GmbH, 1streprint edition, 2013.

[95] Albert Pujol, Roberto Mosca, Judith Farrés, and Patrick Aloy. Unveiling the role of network and systems biology in drug discovery. *Trends in Pharmacological Sciences*, 31(3):115–123, 2010.

[96] Tomas Radivoyevitch. A two-way interface between limited Systems Biology Markup Language and R. *BMC Bioinformatics*, 5:1–9, 2004.

[97] RCoreTeam and R Development Core Team. R: A Language and Environment for Statistical Computing, 2012.

[98] Hans-Jürgen Reinhardt. *Numerik gewöhnlicher Differentialgleichungen: Anfangs- und Randwertprobleme*. Walter de Gruyter, 2012.

[99] Helene Royo. Programming Tools : Adventures With R. *Nature*, 517(7532):109–110, 2015.

[100] Karl Rupp, Philippe Tillet, Florian Rudolf, Josef Weinbub, Andreas Morhammer, Tibor Grasser, Ansgar Jüngel, and Siegfried Selberherr. ViennaCL—Linear Algebra Library for Multi- and Many-Core Architectures. *SIAM Journal on Scientific Computing*, 38(5):S412–S439, 2016.

[101] Rintaro Saito, Michael E. Smoot, Keiichiro Ono, Johannes Ruscheinski, Peng Liang Wang, Samad Lotia, Alexander R. Pico, Gary D. Bader, and Trey Ideker. A travel guide to Cytoscape plugins. *Nature Methods*, 9(11):1069–1076, 2012.

[102] Thomas Schreiber. Measuring Information Transfer. *Physical Review Letters*, 85(2):461–464, jul 2000.

[103] C E Shannon, W Weaver, and University of Illinois Press. A mathematical theory of communication. *The Bell System Technical Journal*, 27(1):379–423, 1948.

[104] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Beno Schwikowski, and Trey Ideker. Cytoscape: A software Environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.

[105] Tatyana Sharpee, Hiroki Sugihara, Andrei V Kurgansky, Sergei P Rebrik, Michael P Stryker, and Kenneth D Miller. Adaptive filtering enhances information transmission in visual cortex. *Nature*, 439(7079):936–942, 2006.

[106] Karline Soetaert, Thomas Petzoldt, and R. Woodrow Setzer. Solving Differential Equations in R : Package deSolve. *Journal of Statistical Software*, 33(9):1–25, 2010.

[107] Eduardo Sontag, Anatoly Kiyatkin, and Boris N. Kholodenko. Inferring dynamic architecture of cellular networks using time series of gene expression, protein and metabolite data. *Bioinformatics*, 20(12):1877–1886, aug 2004.

[108] C. Spearman. The proof and measurement of association between two things. By C. Spearman, 1904. *The American journal of psychology*, 100(3-4):441–471, 1987.

[109] Richard M Stallman and G C C Developer Community. Using the GNU Compiler Collection (For Version 4.1.1). *Free Software Foundation*, 4(02), 2006.

[110] Daniel A Steck. SDERK90, A Fortran 90 Integrator for Ito Stochastic Differential Equations, 2008.

[111] Wolfhart Totschnig. Bodies and their effects: The stoics on causation and incorporeals. *Archiv fur Geschichte der Philosophie*, 95(2):119–147, 2013.

[112] Chris Toumey. Less is Moore. *Nature Nanotechnology*, 11(1):2–3, jan 2016.

[113] Thai Quang Tung, Taewoo Ryu, Kwang H. Lee, and Doheon Lee. Inferring gene regulatory networks from microarray time series data using transfer entropy. *Proceedings - IEEE Symposium on Computer-Based Medical Systems*, pages 383–388, jun 2007.

[114] Maarten Van der Heijden, Marina Velikova, and Peter J.F. Lucas. Learning Bayesian networks for clinical time series analysis. *Journal of Biomedical Informatics*, 48:94–105, 2014.

[115] Christine Vogel and Edward M. Marcotte. Insights into the regulation of protein abundance from proteomic and transcriptomic analyses. *Nature Reviews Genetics*, 13(4):227–232, 2012.

[116] Michael Waechter, Kay Hamacher, Franziska Hoffgaard, Sven Widmer, and Michael Goesele. Random Permutation on a GPU – Is Your Algorithm Unbiased for n != 2ˆm. In *Journal of the ACM*, pages 2011–2011. Springer, 2011.

[117] Michael Waechter, Kathrin Jaeger, Daniel Thuerck, Stephanie Weissgraeber, Sven Widmer, Michael Goesele, and Kay Hamacher. Using graphics processing units to investigate molecular coevolution. *Concurrency Computation Practice and Experience*, 26(6):1278–1296, 2014.

[118] Philipp Weil, Franziska Hoffgaard, and Kay Hamacher. Estimating sufficient statistics in co-evolutionary analysis by mutual information. *Computational Biology and Chemistry*, 33(6):440–444, 2009.

[119] Steve Weston. doParallel: Foreach parallel adaptor for the parallel package. *Analytics, Revolution*, 1(8), 2014.

[120] Michael Wibral, Joseph T. Lizier, Sebastian Vögler, Viola Priesemann, and Ralf Galuske. Local active information storage as a tool to understand distributed neural information processing. *Frontiers in Neuroinformatics*, 8(January):1, jan 2014.

[121] Norbert Wiener. The theory of prediction. *Modern mathematics for the engineer, editor E.F. Beckenbach*, 58(v. 1):165–190, 1956.

[122] Z. Xie and D. Kulasiri. Modelling of circadian rhythms in Drosophila incorporating the interlocked PER/TIM and VRI/PDP1 feedback loops. *Journal of Theoretical Biology*, 245(2):290–304, mar 2007.

[123] Kun Yang, Wenzhe Ma, Huanhuan Liang, Qi Ouyang, Chao Tang, and Luhua Lai. Dynamic simulations on the arachidonic acid metabolic network. *PLoS Computational Biology*, 3(3):0523–0530, mar 2007.

[124] Michael W. Young and Steve A. Kay. Time zones: A comparative genetics of circadian clocks. *Nature Reviews Genetics*, 2(9):702–715, 2001.

[125] Melanie N. Zeilinger, Eva M. Farré, Stephanie R. Taylor, Steve A. Kay, and Francis J. Doyle. A novel computational model of the circadian clock in Arabidopsis that incorporates PRR7 and PRR9. *Molecular Systems Biology*, 2:58, 2006.

Publikationen

- Sebastian Bremm, Tobias Schreck, Patrick Boba, Stephanie Held, Kay Hamacher (2010) Computing and visually analyzing mutual information in molecular co-evolution *BMC Bioinformatics*

- Patrick Boba, Philipp Weil, Franziska Hoffgaard, Kay Hamacher (2010) Co-Evolution in HIV Enzymes *Proc. of BIOINFORMATICS 2010*

- Patrick Boba, Philipp Weil, Franziska Hoffgaard, Kay Hamacher (2011) Intra- and inter-molecular coevolution: The case of HIV1 protease and reverse transcriptase *Communications in Computer and Information Science*

- Patrick Boba, Stefan Gries, Kay Hamacher (2014) R as an Integration Tool in High Performance Computing – Lessons Learned. *Lecture Notes in Informatics (LNI)*

- Patrick Boba, Dominik Bollmann, Daniel Schoepe, Nora Wester, Jan Wiesel und Kay Hamacher (2015) Efficient computation and statistical assessment of transfer entropy. *Frontiers in Physics*

- Patrick Boba und Kay Hamacher (2017) TransferEntropyPT: An R Package to Assess Transfer Entropies via Permutation Tests. *Lecture Notes in Computer Science*

Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit entsprechend den Regeln guter wissenschaftlicher Praxis selbstständig und ohne unzulässige Hilfe Dritter angefertigt habe.

Sämtliche aus fremden Quellen direkt oder indirekt übernommenen Gedanken sowie sämtliche von Anderen direkt oder indirekt übernommenen Daten, Techniken und Materialien sind als solche kenntlich gemacht. Die Arbeit wurde bisher bei keiner anderen Hochschule zu Prüfungszwecken eingereicht.

Darmstadt, den 17.06.2019

Patrick Boba

Danksagung

Abschließend möchte ich ein paar Worte an diejenigen richten, die mich beim Anfertigen dieser Arbeit und während meiner Zeit als Doktorand unterstützt haben.

Zuerst möchte ich meinem Doktorvater Kay Hamacher danken, der mir die Promotion in seiner Arbeitsgruppe ermöglicht hat. Nicht nur die intensiven fachlichen Diskussionen, sondern auch der Blick über den Tellerrand, waren eine Bereicherung meiner Zeit am Fachbereich. Ebenso danken möchte ich meinem Zweitgutachter Gerd Thiel, der sich nicht von der längeren Reifezeit dieser Arbeit hat abschrecken lassen, sondern immer hilfsbereit war.

Der AG danke ich für die wunderbare Zeit, die ja bereits vor dem Beginn meiner Doktorarbeit begonnen hat. Unvergessliche Diskussionen, wie die „wissenschaftlich“ korrekte Unterscheidung zwischen Kuchen und Torte, werde ich immer in froher Erinnerung behalten. Insbesondere Steffi, Sabine und Frank sollen hier namentlich erwähnt sein, da sie mich bei der Anfertigung dieser Arbeit noch weit über ihre Zeit in der AG motiviert haben und mich tatkräftig bei der Korrektur unterstützten.

Auch außerhalb des Mikrokosmos "Computational Biology"gibt es Personen denen ich danken möchte: Manni, weil wir uns gegenseitig durchs Studium gehievt haben. Steffen und Julia, die viel mehr über die Entwicklung von bioinformatischen Tools hören mussten, als sie wohl gehofft hatten.

Meinen Eltern und Geschwistern möchte ich danken, dass sie mir ermöglicht haben meinen eigenen (nicht immer geradlinigen) Weg einzuschlagen und ich mir ihrer Hilfe immer sicher sein kann. Und vor allem möchte ich Kathrin danken, die den Mut hatte, kritisch zu sein, wenn sie musste, und mich dabei dennoch bedingungslos unterstützt und angespornt hat.

Danke.