

# **Probabilistische Methoden zur Rekonstruktion biologischer Netzwerke**

Prof. Dr. Sven Rahmann

LS 11, Fakultät für Informatik, TU Dortmund

Spezialvorlesung Sommersemester 2009  
Spezialvorlesung Wintersemester 2010/11  
ENTWURF VOM 12. JANUAR 2011



---

# Inhaltsverzeichnis

---

<b>1</b>	<b>Modelle für Netzwerke</b>	<b>1</b>
1.1	Beispiele für Netzwerke . . . . .	1
1.2	Graphen als Modelle für Netzwerke . . . . .	2
1.3	Eigenschaften von Netzwerken . . . . .	3
1.3.1	Verteilung der Knotengrade . . . . .	3
1.3.2	Dichtemaße . . . . .	5
1.3.3	Netzwerk-Motive . . . . .	6
1.4	Zufällige Graphen . . . . .	6
<b>2</b>	<b>Transkriptionelle Genregulation</b>	<b>9</b>
2.1	Einleitung . . . . .	9
2.2	Modelle für Transkriptionsfaktorbindestellen . . . . .	9
<b>3</b>	<b>Motivsuche mit dem EM-Algorithmus</b>	<b>13</b>
3.1	Ein likelihood-basierter Ansatz zur Motivsuche mit PFMs . . . . .	13
3.2	Der EM-Algorithmus für Mehrkomponentenmodelle . . . . .	15
3.3	Motivsuche mit dem EM-Algorithmus . . . . .	20
<b>4</b>	<b>Identifizierbarkeit in Signal-Sensor-Netzwerken</b>	<b>23</b>
4.1	Motivation . . . . .	23
4.2	Identifizierbarkeit . . . . .	26
4.3	Optimale Sensoren-Auswahl . . . . .	29
4.4	Separable Signale . . . . .	33
<b>5</b>	<b>Analyse von Reaktionsnetzwerken</b>	<b>35</b>
5.1	Einleitung . . . . .	35
5.2	Die stöchiometrische Matrix . . . . .	35
5.3	Elemente und Erhaltungsgrößen . . . . .	37
5.4	Fluxe und Reaktionsdynamik . . . . .	38
5.5	Vier durch die stöchiometrische Matrix definierte Unterräume . . . . .	39

*Inhaltsverzeichnis*

5.6	Interne und Externe Reaktionen . . . . .	40
5.7	Extreme Pathways . . . . .	41
5.8	Abstraktion: Double Description Method . . . . .	44
<b>A</b>	<b>Matrixalgebra</b>	<b>49</b>
A.1	Grundlagen . . . . .	49
A.2	Eigenwerte . . . . .	50
A.3	Orthogonale Matrizen . . . . .	52
A.4	Die Singulärwertzerlegung . . . . .	53
A.5	Anwendungen der Singulärwertzerlegung . . . . .	54
A.5.1	Least-Squares-Lösung überbestimmter Gleichungssysteme . . . . .	54
A.5.2	Die Pseudoinverse einer Matrix . . . . .	55
A.5.3	Die beste Rang- $k$ -Approximation einer Matrix . . . . .	56

---

## Modelle für Netzwerke

---

Eine Struktur aus einer Menge von Objekten zusammen mit der Information, welche dieser Objekte jeweils in einer Beziehung zueinander stehen, nennt man *Netzwerk*. Wir geben einige Beispiele in Abschnitt 1.1. Sind die Beziehungen symmetrisch, modelliert man ein Netzwerk normalerweise mit Hilfe eines (einfachen) ungerichteten Graphen. Sind die Beziehungen nicht notwendig symmetrisch, modelliert man ein Netzwerk mit einem gerichteten Graphen (engl. directed graph, digraph). Die grundlegenden Begriffe zu Graphen werden in Abschnitt 1.2 wiederholt. Im Anschluss (Abschnitt 1.3) beschreiben wir einige globale Eigenschaften eines Netzwerks, etwa die Verteilung der Knotengrade, sowie Dichtemaße oder das Auftreten bestimmter “Motive”. Häufig ist es interessant, solche beobachteten Eigenschaften mit denen eines “zufälligen Graphen” zu vergleichen. Dazu stellen wir in Abschnitt ?? verschiedene Zufallsmodelle für Graphen vor.

### 1.1 Beispiele für Netzwerke

Die Objekte und deren Beziehungen können zahlreiche verschiedene Dinge sein. Wie wir sehen werden, lassen sich viele biologische Prozesse durch Netzwerke darstellen:

- Netzwerke transkriptioneller Regulation (Transkriptionsfaktoren regulieren Zielgene, siehe Abschnitt ??),
- Protein-Interaktionsnetzwerke (Proteine interagieren physikalisch miteinander und bewirken etwas),
- Metabolische Netzwerke (Metabolite reagieren in biochemischen Reaktionen zu anderen Metaboliten),
- Signaltransduktionsnetzwerke (“Information” verbreitet sich in der Zelle),

## FEHLT: Graph

Abbildung 1.1: Zwei Graphen. Links: ungerichteter Graph. Rechts: gerichteter Graph.

- (Biochemische) Reaktionsnetzwerke (Oberbegriff).

Ein weiteres Beispiel ist ein Netzwerk von Personen mit der Beziehung “kennt”; man kann also sagen “Person A kennt Person B”. Solche sozialen Netzwerke sind in den letzten Jahren durch Plattformen wie Xing, Facebook und StudiVZ ins öffentliche Interesse gerückt. Ein anderes Beispiel ist das WWW, in dem Webseiten durch die Beziehung “enthält einen Link auf” miteinander verknüpft sind.

## 1.2 Graphen als Modelle für Netzwerke

Obwohl die obigen Beispiele schon deutlich machen, wie verschieden Netzwerke sein können, gibt es oft erstaunliche Gemeinsamkeiten, die es erlauben, dieselben oder ähnliche Methoden zur ihrer Analyse anzuwenden. In diesem Kapitel geht es um solche allgemeinen Eigenschaften von Netzwerken. Mathematisch werden Netzwerke durch *Graphen* modelliert. Deshalb beginnen wir mit einer kurzen Wiederholung der wichtigsten Definitionen der Graphentheorie.

**1.1 Definition** (Ungerichteter Graph). Ein *einfacher ungerichteter Graph* ist ein Tupel  $(V, E)$  mit  $E \subseteq \binom{V}{2}$ ; d. h.  $E$  ist eine Teilmenge aller zweielementigen Teilmengen von  $V$ . Wir bezeichnen ein  $v \in V$  als *Knoten* (engl. vertex) und ein  $e \in E$  als *Kante* (engl. edge).

Insbesondere kommen per Definition in einem einfachen ungerichteten Graphen keine Schleifen vor (Kanten von  $v$  nach  $v$ ).

**1.2 Definition** (Gerichteter Graph). Ein *gerichteter Graph* ist ein Tupel  $(V, E)$  mit  $E \subseteq V \times V$ ; d. h.  $E$  ist eine Teilmenge aller geordneten Paare von Elementen aus  $V$ .

Beim Zeichnen von Graphen werden Knoten oft als Kreise und Kanten als Verbindungslinien zwischen den Kreisen dargestellt. Ist ein Graph gerichtet, so haben die Verbindungen eine Richtung, die durch einen Pfeil angezeigt wird. Abbildung 1.1 enthält je ein Beispiel für einen gerichteten und einen ungerichteten Graphen. Im Zusammenhang mit Graphen gibt es einige Begriffe, die man sich aneignen sollte.

**1.3 Definition** (Begriffe zu Graphen). Ein (gerichteter) *Pfad der Länge  $k$*  von einem Knoten  $u$  zu einem Knoten  $v$  ist eine Sequenz von Knoten  $(u =: v_0, v_1, \dots, v_{k-1}, v_k := v)$  mit  $v_i \in V$  für  $0 \leq i \leq k$  und  $(v_i, v_{i+1}) \in E$  für  $0 \leq i < k$ . Analog ist ein ungerichteter Pfad der Länge  $k$  in ungerichteten Graphen definiert. Man sagt, Knoten  $v$  ist *erreichbar* von Knoten  $u$ , wenn es einen Pfad von  $u$  nach  $v$  gibt. Ein ungerichteter Graph heißt *zusammenhängend*, wenn jeder Knoten von jedem erreichbar ist. Ein gerichteter Graph heißt *stark zusammenhängend*, wenn jeder Knoten von jedem erreichbar ist und *schwach zusammenhängend*, wenn der zugehörige ungerichtete Graph (durch Vergessen der Kantenrichtung) zusammenhängend ist.

Ein Graph heißt *zyklisch* wenn er einen Knoten  $v$  beinhaltet, für den es einen Pfad einer Länge  $\geq 1$  von  $v$  nach  $v$  gibt; andernfalls heißt er *azyklisch* oder *kreisfrei*.

Gegeben zwei Graphen  $G = (V, E)$  und  $G' = (V', E')$ , so heißt  $G'$  *Subgraph* von  $G$  wenn  $V' \subseteq V$  und  $E' \subseteq E$ . Falls zu gegebenem  $V'$  genau  $E' = \{(v_1, v_2) \in E \mid v_1, v_2 \in V'\}$  ist, so heißt  $G'$  der von  $V'$  *induzierte Subgraph* von  $G$ ; er enthält alle zwischen den Knoten aus  $V'$  möglichen Kanten aus  $E$ .

Die Graphen  $G$  und  $G'$  heißen *isomorph*, wenn es eine bijektive Abbildung  $\phi : V \rightarrow V'$  gibt, so dass  $(v_1, v_2) \in E$  genau dann gilt, wenn  $(\phi(v_1), \phi(v_2)) \in E'$ . (Es kann schwierig sein, Isomorphie zu entscheiden und  $\phi$  zu finden.)

Der *Grad* eines Knotens  $v$  in einem ungerichteten Graphen ist die Zahl der Kanten, die diesen Knoten enthalten, also formal  $|\{e \in E \mid v \in e\}|$ . Die *Nachbarschaft* von  $v$  ist  $\{v' \in V \mid \{v, v'\} \in E\}$ ; der Grad ist also die Größe der Nachbarschaft. Bei einem gerichteten Graphen unterscheiden wir den *Eingangsgrad* oder *Eingrad*  $|\{(v_1, v_2) \in E \mid v_2 = v\}|$  und den *Ausgangsgrad* oder *Ausgrad*  $|\{(v_1, v_2) \in E \mid v_1 = v\}|$ .

Sowohl Knoten als auch Kanten eines Graphen können mit zusätzlichen Informationen annotiert sein. Zum Beispiel könnte man in einem sozialen Netzwerk mit der Beziehung “kennt” (s.o.) an jede Kante schreiben, seit wann die Personen sich kennen. Genauso könnte man jeden Knoten mit der Haarfarbe der Person annotieren.

## 1.3 Eigenschaften von Netzwerken

Ist ein Graph gegeben, kann man verschiedener seiner Eigenschaften beschreiben, beispielsweise die Verteilung seiner Knotengrade oder verschiedene Dichteparameter.

### 1.3.1 Verteilung der Knotengrade

Ein interessantes Merkmal von Netzwerken ist die Verteilung der Knotengrade. Erstaunlicherweise stellt man fest, dass die Verteilung der Knotengrade vieler biologischer Netzwerke näherungsweise einem *Potenzgesetz* (engl. power law) folgt (siehe ?). Das bedeutet

$$N(d) = c \cdot d^{-z} + o(d^{-z}),$$

wobei  $N(d)$  die Anzahl der Knoten mit Grad  $d$  und  $c > 0$ ,  $z > 0$  Konstanten sind. Das bedeutet, dass die allermeisten Knoten einen kleinen Grad haben, während eine kleine Zahl von Knoten einen sehr hohen Grad hat. Solche stark verbundenen Knoten nennt man *hubs*.

Es gilt

$$\begin{aligned} N(d) &= c \cdot d^{-z} = c \cdot e^{-z \cdot \log d} \\ \iff \log N(d) &= \log c - z \cdot \log d \\ &= -z \cdot \log d + c' \end{aligned}$$

Daher müssen wir nach einem linearen Zusammenhang der logarithmierten Knotengrade mit den logarithmierten Anzahlen suchen. Oder anders gesagt: In einem Histogramm, bei dem X- und Y-Achse logarithmisch skaliert sind (“doppelt logarithmische Darstellung”), muss sich eine Gerade ergeben.

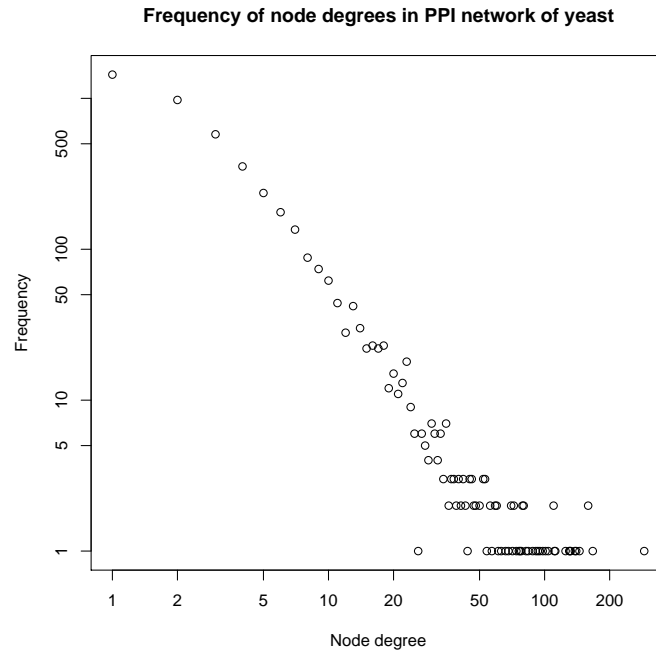


Abbildung 1.2: Gradverteilung der Knoten im Protein-Protein-Interaktionsnetzwerk der Hefe *Saccharomyces cerevisiae* aus der MIPS-Datenbank

In der Praxis kann es schwierig sein, den Zusammenhang genau nachzuweisen, denn erstens sind die Daten (wie Anzahlen und Grade) endlich, so dass die Asymptotik nicht greift; zweitens sind sie häufig diskret, so dass es beispielsweise keine möglichen Zahlen zwischen 0 und 1 gibt; drittens kann man (was auch durch den  $o()$ -Term ausgedrückt wird), kein exaktes Potenzgesetz erwarten; viertens ist unklar, wie man die Abweichung der Punkte von einer Geraden im doppelt logarithmischen Plot quantifizieren und ab wann man sich gegen eine approximative Gerade entscheiden soll.

In der Praxis kann man die bestmögliche Gerade und damit auch die Parameter  $c'$  und  $z$  mittels linearer Regression bestimmen (Minimierung der Fehlerquadrate), aber auch hier gibt es verschiedene Alternativen, beispielsweise eine robuste lineare Regression oder noch speziellere Verfahren ?.

Wir wollen die These eines Potenzgesetzes für das Beispiel des Netzwerks der Protein-Protein-Interaktionen der Bäckerhefe (*Saccharomyces cerevisiae*) aus der CYGD-Datenbank (Comprehensive Yeast Genome Database) des MIPS (Munich Information Center for Protein Sequences)<sup>1</sup> überprüfen. Abbildung 1.2 zeigt die Gradverteilung. Lässt man Knoten mit sehr kleinem Grad (etwa 1) und Knoten mit solchen Graden, die nur sehr selten auftauchen (etwa nur 1 mal) weg, erkennt man relativ gut eine Gerade im doppelt logarithmischen Plot, deren Parameter man ablesen kann.

**Verwendung kumulierter Häufigkeiten.** Da oftmals nicht alle Grade vorkommen und die Häufigkeit stark schwankt, kann es zur robusteren Bestimmung des Potenzgesetzes sinn-

<sup>1</sup><ftp://ftpmips.gsf.de/yeast/PPI/>

voll sein, nicht die Häufigkeiten  $N(d)$ , sondern die kumulierten Häufigkeiten  $N^+(d) := \sum_{\delta \leq d} N(\delta)$  zu betrachten. Durch Integration erhält man approximativ

$$\begin{aligned} N(d) &= c \cdot d^{-z} \\ N^+(d) &= \gamma \cdot d^{-z+1} \\ \log N^+(d) &= (-z + 1) \log d + \gamma' \end{aligned}$$

mit einem anderen Faktor  $\gamma$  bzw.  $\gamma' = \log \gamma$ . Hierbei ist  $z > 1$  vorausgesetzt.

### 1.3.2 Dichtemaße

Oft ist es von Interesse, wie dicht ein Netzwerk (oder induziertes Teilnetzwerk) verbunden ist. Solche Kennzahlen nennt man *Dichtemaße*. Wir betrachten vorerst nur einfache ungerichtete Graphen. Zur Abkürzung schreiben wir  $uv$  für die Kante  $\{u, v\}$ .

**Kantendichte.** Ein einfaches Beispiel für ein Dichtemaß eines Graphen  $(V, E)$  ist seine *Kantendichte*, also der Anteil der vorhandenen Kanten an allen möglichen Kanten:  $|E|/\binom{|V|}{2}$ . Die Kantendichte liegt zwischen 0 und 1. Sind alle  $\binom{n}{2}$  möglichen Kanten zwischen  $n$  Knoten vorhanden, dann nennt man diesen Graphen *vollständig* oder eine *Clique*.

**Globaler Clusteringkoeffizient.** Ein weiteres Dichtemaß ergibt sich über Knotentripel. Wir schreiben  $uvw$  für  $\{u, v, w\}$ .

Ein Knotentripel kann 0, 1, 2, oder 3 Kanten haben. Das Tripel  $uvw$  ist zusammenhängend, wenn es mindestens zwei Kanten zwischen den drei Knoten gibt. Wir nennen das Tripel  $uvw$  ein *Dreieck*, wenn alle drei Kanten vorhanden sind.

Als (globales) Dichtemaß bietet sich an: Anzahl der Dreiecke relativ zur Anzahl der zusammenhängenden Tripel. Es heißt *globaler Clusteringkoeffizient*. Dieses Maß lässt sich sogar sinnvoll auf nicht zusammenhängenden Graphen anwenden und quantifiziert, wie weit ein Graph davon entfernt ist, eine disjunkte Vereinigung von Cliques zu sein.

**1.4 Definition.** Ein Graph heißt *transitiv*, wenn für alle Knotentripel  $uvw$  gilt: Aus  $uv \in E$  und  $vw \in E$  folgt auch  $uw \in E$ . Äquivalent: Alle zusammenhängenden Tripel sind Dreiecke.

Man kann Folgendes beweisen:

**1.5 Lemma.** *Ein einfacher ungerichteter Graph ist genau dann eine disjunkte Vereinigung von Cliques, wenn er transitiv ist.*

Wir geben noch folgenden Ausblick auf einen Clustering-Algorithmus: Will man Objekte in disjunkte Gruppen einordnen ("clustern"), und hat Beziehungen zwischen den Objekten als Eingabe gegeben, dann kann man denjenigen transitiven Graphen suchen, der aus dem Eingabegraphen durch möglichst wenig Kantenänderungen (Hinzufügungen, Löschungen) hervorgeht. Dies ist das sogenannte *Cluster-Editing-Problem*. Es ist NP-schwer, aber in der Praxis für nicht zu große Graphen gut lösbar. Die resultierenden Cliques bilden die gesuchten Cluster.

**Durchschnittlicher Clusterkoeffizient.** Man kann für jeden Knoten  $v$  zunächst einen *lokalen Clusterkoeffizienten* definieren. Dies sei die Kantendichte in dem Graphen, der durch die Nachbarschaft von  $v$  (ohne  $v$  selbst) induziert wird. (Alternativ kann man auch den globalen Clusterkoeffizienten der Nachbarschaft von  $v$  nehmen; man erhält so verschiedene Varianten.)

Der *durchschnittliche Clusterkoeffizient* eines Graphen ist dann der Durchschnitt der lokalen Clusterkoeffizienten über alle Knoten.

Man veranschaulicht sich dies am besten an einem nicht-trivialen Beispiel mit 5 Knoten, in denen einige Kanten fehlen.

### 1.3.3 Netzwerk-Motive

Netzwerk-Motive sind häufig vorkommende isomorphe induzierte Subgraphen eines Netzwerks. In Biologischen Netzwerken sind insbesondere als sinnvoll bekannt:

- Feed-forward loop (konsistent, wenn sich die Kantenbeschriftungen nicht widersprechen; inkonsistent sonst)
- Single-input motif
- Multiple-input motif

Komplexere Motive mag es ebenfalls geben, aber diese lassen sich nicht so einfach interpretieren.

## 1.4 Zufällige Graphen

Es gibt verschiedene Modelle für zufällige Graphen, die verschiedene Eigenschaften haben. Wir untersuchen diese Modelle, um zu quantifizieren, wie überraschend beobachtete Eigenschaften in echten Graphen sind. Wenn wir beispielsweise in einem Proteinnetzwerk mit 4500 Knoten genau 170 Vierercliquen beobachten, können wir uns fragen, ob das überraschend viele oder wenige sind, oder etwas, was man “durch Zufall” erwarten würde. Dazu muss man natürlich genau sagen, was man unter einem zufälligen Graphen versteht. Dann kann man durch Simulation (oder auch mit Mathematik) die Wahrscheinlichkeit berechnen, dass mindestens 170 Vierercliquen auftreten. Ist diese Wahrscheinlichkeit sehr gering, die die Beobachtung überraschend und hat möglicherweise eine biologische Bedeutung.

**Gilbert-Modell.** Man gibt eine Knotenanzahl  $n$  und eine durchschnittliche Kantendichte  $0 \leq p \leq 1$  vor. Jede der  $\binom{n}{2}$  möglichen Kanten existiert unabhängig von den anderen mit Wahrscheinlichkeit  $p$ . Die Anzahl der Kanten ist also binomialverteilt mit den Parametern  $\binom{n}{2}$  und  $p$ . Da die Kanten unabhängig voneinander sind, ist ein Graph in diesem Modell mit Hilfe eines Zufallszahlengenerators sehr einfach zu simulieren. Man kann das Modell leicht auf gerichtete Graphen erweitern. Man kann auch mit Hilfe einer  $n \times n$ -Wahrscheinlichkeitsmatrix  $(p_{ij})$  für jede Kante  $(i, j)$  eine individuelle Wahrscheinlichkeit vorgeben.

**Erdős-Rényi-Modell.** Man gibt eine Knotenanzahl  $n$  und eine exakte Kantenanzahl  $m$  vor. Unter allen Graphen mit  $n$  (durchnummerierten und daher unterscheidbaren) Knoten und  $m$  Kanten soll ein zufälliger gewählt werden. Viele Eigenschaften eines solchen Graphen sind für große  $n$  sehr ähnlich zu einem zufälligen Graphen aus dem Gilbert-Modell mit  $p := m/n$ , aber die Modelle sind nicht identisch.

Man kann einen solchen Graphen konstruieren, indem ihn als Realisierung eines Prozesses betrachtet, der mit  $n$  Knoten und keiner Kante beginnt und dann in  $m$  Schritten in jedem Schritt eine zufällige noch nicht existierende Kante hinzufügt.

Wie kann man dies effizient realisieren? Man kann wiederholt zufällige Kanten aus den  $\binom{n}{2}$  möglichen ziehen. Existiert die Kante noch nicht, wird sie hinzugefügt. Existiert sie schon, passiert nichts. Dies wird solange wiederholt, bis  $m$  Kanten existieren. Diese Methode ist effizient, solange  $m/n$  sehr klein ist, da dann die "Misserfolge" nicht ins Gewicht fallen. Ist aber  $m \approx n$ , muss man gegen Ende sehr viele Versuche aufwenden, um eine weitere Kante hinzuzufügen. Man könnte statt dessen dann zufällig die  $n - m$  nicht vorhandenen Kanten ziehen.

Mathematiker (insbesondere Erdős und Rényi) haben dieses Modell in sehr vielen Details untersucht. Wir geben nur ein Beispiel: Ein interessante Frage ist, wie man  $p$  wählen muss, damit der Graph mit großer Wahrscheinlichkeit zusammenhängend ist. Eine mathematische Aussage hierzu ist:

**1.6 Satz.** Sei  $\kappa(G)$  die Zusammenhangszahl des Graphen, also die minimale Anzahl an Kanten, die man mindestens löschen muss, damit der Graph nicht mehr zusammenhängend ist (ein nicht zusammenhängender Graph  $G$  hat  $\kappa(G) = 0$ ). Es sei  $n \rightarrow \infty$  und  $m(n) := \frac{1}{2}n(\log n + k \log \log n + x + o(1))$ . Dann ist

$$\kappa(\mathcal{G}(n, m)) = k \rightarrow 1 - \exp(-e^{-x}/k!).$$

**Barabási-Albert-Modell.** Man gibt eine Knotenanzahl  $n$  vor. Das Netzwerk wird in einem zufälligen Prozess in  $n - 2$  Schritten generiert. Ausgangspunkt ist eine 2-Clique. In Schritt  $t$  wird Knoten  $t + 2$  hinzugefügt und über genau eine Kante mit dem existierenden Netzwerk verbunden. Mit welchem Knoten der neue Knoten verbunden wird, hängt von der existierenden Gradverteilung ab.

Sei  $d_i$  der Grad des Knoten  $i$  zum aktuellen Zeitpunkt. Die Wahrscheinlichkeit, dass die Kante  $\{i, t + 2\}$  gezogen wird, ist  $d_i/D$  mit  $D := \sum_{i < t+2} d_i$ . Knoten, die schon einen hohen Grad haben, bekommen also bevorzugt weitere Nachbarn. Das Prinzip ist auch bekannt als *preferential attachment*, manchmal auch mit "the rich get richer" umschrieben. Die genannte Kombination auf Wachstum und preferential attachment erzeugt Netzwerke, deren Gradverteilung einem Potenzgesetz mit Exponent  $-3$  gehorcht.

Eigenschaften:

- Das Netzwerk ist verbunden (einfacher Induktionsbeweis).
- Die Gradverteilung der Knoten folgt einem Potenzgesetz  $N(d) \sim d^{-3}$ .

Varianten:

- Man beginnt mit einem anderen Kernnetzwerk als einer 2-Clique.

## 1 Modelle für Netzwerke

- Für jeden neuen Knoten wird mehr als eine Kante gezogen.
- Bei der Berechnung der Wahrscheinlichkeiten werden Pseudocounts zum Knotengrad addiert. Man erhält so andere Exponenten im Potenzgesetz der Gradverteilung.
- Die Wahrscheinlichkeit, den Zielknoten  $i$  für die neue Kante auszuwählen, wird nicht proportional zu  $d_i$ , sondern proportional zu  $d_i^q$  mit  $q > 0$  gewählt.

Man kann durch Simulation bestimmen, welche Eigenschaften so ein Netzwerk hat.

TODO: Skaleninvarianz small-world network

---

# Transkriptionelle Genregulation

---

## 2.1 Einleitung

Promotor

## 2.2 Modelle für Transkriptionsfaktorbindestellen

Die DNA-Sequenz an den Bindestellen (BS) eines Transkriptionsfaktors (TF) ist für diesen TF relativ charakteristisch; d.h. man kann weitere Bindestellen vorhersagen, wenn man einige davon kennt. Dazu benötigt man ein Modell der Bindestelle.

Wir betrachten folgende Situation: Es sind bereits einige Bindestellen eines Transkriptionsfaktors bekannt und experimentell verifiziert. Alle beobachteten Sequenzen sind gleich lang und aligniert (einander entsprechende Positionen stehen untereinander). Welche Modelle bieten sich an, und wie können wir damit jeweils nach weiteren BS suchen?

**Menge.** Das Modell ist eine String-Menge, die die beobachteten Sequenzen an Bindestellen aufzählt. Vorteile: Modell einfach zu erstellen. Nachteile: lange Beschreibung (Aufzählung aller Beobachtungen), daher komplex, schlecht visualisierbar, ungeeignet beobachtete TFBS zu verallgemeinern.

**Strikter Consensus.** An jeder Position wird das häufigste Nukleotid bestimmt; der String aus den häufigsten Nukleotiden bildet das Modell. Vorteile: Modell einfach zu erstellen, leicht visualisierbar (ein einzelner einfacher String). Nachteile: Consensus kann von allen beobachteten TFBS-Sequenzen verschieden sein, Information über die Beobachtungen geht fast komplett verloren.

**Generalized String (IUPAC-String).** Ein *verallgemeinerter String* oder *generalized string* über einem Alphabet  $\Sigma$  ist ein String, dessen Symbole nichtleere Teilmengen von  $\Sigma$  sind. Im DNA-Alphabet  $\{A, C, G, T\}$  gibt es 15 nichtleere Teilmengen. Diese bekommen eigene Symbole zur Abkürzung: IUPAC-Code.

Ein verallgemeinerter String  $M = M_1 \dots M_L$  passt zu einem (einfachen) String  $s = s_1 \dots s_L$  der gleichen Länge  $L$ , wenn  $s_i \in M_i$  für alle  $i$ . Ein TFBS-Modell ist der spezifischste IUPAC-String, der zu allen beobachteten TFBS-Instanzen passt.

Vorteile: Modell einfach zu erstellen: Bilde die Vereinigung aller beobachteten Zeichen an jeder Position, leicht visualisierbar (ein einzelner IUPAC-String), verallgemeinert beobachtete TFBS-Instanzen

Nachteile: kann sehr stark verallgemeinern, passt evtl. auch auf sehr viele Strings, die nicht beobachtet wurden, Horizontale Abhängigkeiten in den TFBS-Instanzen gehen verloren.

**Menge von IUPAC-Strings.** Kompaktere Darstellung der einfachen Menge; ähnliche Strings können zu einem IUPAC-String zusammengefasst werden. Ziel ist aber wie bei der Menge eine exakte Repräsentation der TFBS-Instanzen. Algorithmisches Problem: Finde möglichst wenig IUPAC Strings, die genau zur Menge der beobachteten Instanzen passen (nicht zu mehr, nicht zu weniger).

Vorteile und Nachteile: wie bei Menge, aber kompaktere Darstellung.

**(Menge von) (verallgemeinerter/n) String(s) mit Fehlerumgebung.** Die Hamming-Distanz zwischen zwei (einfachen) Strings ist die Zahl der Positionen, an denen sie sich unterscheiden. Die (Hamming-)  $d$ -Nachbarschaft eines Strings  $s$  ist die Menge aller Strings, die höchstens den Abstand  $d$  zu  $s$  haben. Die  $d$ -Nachbarschaft eines verallgemeinerten Strings  $M$  ist die Menge aller Strings, die höchstens den Abstand  $d$  zu einem String haben, der zu  $M$  passt. Die  $d$ -Nachbarschaft einer Menge von (verallgemeinerten) Strings ist die Menge aller Strings, die in der  $d$ -Nachbarschaft mindestens eines Strings aus der Menge liegen.

Das Modell besteht aus einer Menge  $M$  von IUPAC-Strings und einer Fehlerschranke  $d$ . In dieser Darstellung ist sowohl eine exakte Aufzählung möglich als auch eine verallgemeinernde Darstellung. Man muss abwägen:

- Große Menge  $M$ , kleines  $d$ , exakte Repräsentation der beobachteten TFBS (“kompakte Aufzählung”)
- Mittelgroße Menge  $M$ , mittleres  $d$ , ungefähre Repräsentation der TFBS (leichte Verallgemeinerung)
- Kleine Menge  $M$ , großes  $d$ , Repräsentation von mehr Sequenzen als TFBS (starke Verallgemeinerung)

**PSSM oder PWM (Position-Specific Scoring Matrix oder Position Weight Matrix).** Die Idee besteht darin, die horizontalen Abhängigkeiten der beobachteten TFBS zu vergessen, aber ansonsten alle Informationen quantitativ zu erhalten.

Die PCM (Position Count Matrix) zählt an jeder Position die absolute Häufigkeit jedes Symbols. Die PFM (Position Frequency Matrix, auch Profil oder profile) gibt an jeder Position die relative Häufigkeit jedes Symbols an (Wahrscheinlichkeiten). Um Wahrscheinlichkeiten von 0 zu vermeiden, wird vor der Normalisierung i.d.R. eine kleine Konstante zu jedem Eintrag der PCM addiert. Sei  $P_{c,i}$  die Wahrscheinlichkeit von Symbol  $c$  an Position  $i$  in der PFM. Sei  $\pi_c$  die Hintergrundverteilung von Symbol  $c$  (z.B. relative Häufigkeit im Genom oder in der Umgebung der TFBS).

Die PSSM oder PWM  $S$  wird dann berechnet als

$$S_{c,i} := \log_2(P_{c,i}/\pi_c)$$

(Logarithmus des Verhältnisses von positionsspezifischer Häufigkeit zur Hintergrund-Häufigkeit eines Symbols).

Ein stark positiver Wert heißt: Das Symbol kommt in den TFBS-Instanzen deutlich häufiger an dieser Position vor als im Hintergrund. Die Einheit dieser Score-Werte sind Bits (bei Verwendung der Basis 2).

Mit einer PSSM  $S = (S_{c,i})$  kann man jeden String  $s = s_1 \dots s_L$  gleicher Länge bewerten, indem man die positions-spezifischen Scores für  $s$  aufsummiert.

$$Score_S(s) := \sum_{i=1}^L S_{s_i,i}$$

Ein String  $s$  passt zur PSSM  $S$ , wenn  $Score_S(s) \geq T$  für einen noch zu wählenden Schwellenwert (threshold)  $T$ .

Wie sollte  $T$  gewählt werden? So dass die beobachteten TFBS-Instanzen möglichst alle einen Score oberhalb  $T$  erreichen und alle anderen Sequenzen einen Score unterhalb  $T$  erreichen. Dies wird in der Regel nicht möglich sein, und man muss einen Kompromiss treffen (wie viele falsch positive und falsch negative Strings lasse ich zu?). Eine geringe Anzahl zusätzlicher Strings oberhalb  $T$  ist sogar erwünscht, damit das Modell die gemachten Beobachtungen verallgemeinert.

PWMs lassen sich mit Sequenzlogos visualisieren. Ein Sequenzlogo zu einer PWM der Länge  $L$  besteht aus  $L$  Symboltürmen. Die Höhe des  $i$ -ten Turms ist der Informationsgehalt der  $i$ -ten Spalte der PFM; die relative Höhe jedes Symbols in einem Turm entspricht seiner relativen Häufigkeit.

Der Informationsgehalt berechnet sich als relative Entropie der positionsspezifischen Verteilung zur Hintergrundverteilung (in Bits); häufig wird hier einfach die Gleichverteilung als Hintergrundverteilung  $\pi$  gewählt:

$$H(P_i||\pi) = \sum_c P_{c,i} \cdot \log_2 \frac{P_{c,i}}{\pi_c}$$

## 2 *Transkriptionelle Genregulation*

Alternative Interpretation: Durchschnittlicher Score-Wert, gewichtet mit der positionsspezifischen Symbolverteilung. Man kann beweisen: Die relative Entropie zwischen zwei Verteilungen ist stets nichtnegativ. Sie ist genau dann gleich null, wenn die Verteilungen gleich sind.

**Fazit.** PWMs haben sich als Modell durchgesetzt, obwohl sie nicht unbedingt für jeden TF die beste Motivbeschreibung liefern. Sie modellieren allerdings gut die Tatsache, dass manche Positionen wichtiger sind als andere.

---

## Motivsuche mit dem EM-Algorithmus

---

### 3.1 Ein likelihood-basierter Ansatz zur Motivsuche mit PFMs

Wir spezifizieren das allgemeine Motivsuche-Problem wie folgt.

**3.1 Problem** (likelihood-basierte Motivsuche mit PFMs). Gegeben sind  $N$  Sequenzen  $y = (y_1, \dots, y_N)$  endlicher Länge über einem Alphabet  $A = \{a_1, \dots, a_K\}$ , sowie eine Motivlänge  $L$ .

Gesucht sind *auffällige Motive* in  $y$ . Bekanntlich müssen wir präzisieren, was Motive sind und was wir unter “auffällig” verstehen.

In diesem Abschnitt verstehen wir unter einem Motiv eine  $K \times L$ -Positions-Häufigkeitsmatrix (position frequency matrix, PFM): In jeder der  $L$  Spalten steht eine Wahrscheinlichkeitsverteilung über dem Alphabet.

Zum Bewerten eines Motivs benutzen wir einen likelihood-Ansatz; d.h. wir suchen das Motiv, das die likelihood des Gesamtmodells maximiert. •

Wir beschreiben jetzt das Modell.

Zur Vereinfachung stellen wir uns vor, dass wir nicht  $N$  Sequenzen gegeben haben, sondern direkt deren Teilstrings der Länge  $L$ , die wir ab jetzt als unabhängig betrachten. Die Eingabe besteht also aus  $x = (x_1, \dots, x_n)$  mit  $x_i = (x_{i1}, \dots, x_{iL}) \in A^L$  für alle  $1 \leq i \leq n$ .

Wir stellen uns vor, dass jedes  $L$ -mer  $x_i$  entweder vom Motiv-Modell (der PFM) oder von einem Hintergrundmodell (einer nicht positionsspezifischen Verteilung) wie folgt erzeugt wird. Zunächst wird für jedes  $x_i$  die Modell-Komponente (Motiv oder Hintergrund) ausgewählt. Mit Wahrscheinlichkeit  $\lambda_1$  wird das Motiv gewählt; mit Wahrscheinlichkeit  $\lambda_2 = 1 - \lambda_1$  der

### 3 Motivsuche mit dem EM-Algorithmus

Hintergrund. Diese Entscheidung fassen wir in Indikatorvariablen  $Z_{i1}, Z_{i2}$  zusammen; es sei  $Z_{ij} := 1$ , wenn für  $x_i$  Komponente  $j$  gewählt wurde, und  $Z_{ij} := 0$  sonst.

Die Parameter der beiden Komponenten bezeichnen wir mit  $\theta_1$  bzw.  $\theta_2$ . Wurde das Motivmodell gewählt, so werden  $L$  Symbole nach der PFM  $\theta_1 = f = (f_{ak})$  gezogen, das  $k$ -te Symbol nach der  $k$ -ten Spalte. Die Wahrscheinlichkeit, ein  $L$ -mer  $x_i$  zu erzeugen, ist damit

$$\mathbb{P}_{\theta_1}(x_i) = \prod_{k=1}^L f_{x_{ik},k}. \quad (3.1)$$

Wurde das Hintergrundmodell gewählt, so werden  $L$  Symbole nach der Hintergrundverteilung  $\theta_2 = f_0 = (f_{a0})$  gezogen. Die Wahrscheinlichkeit, ein  $L$ -mer  $x_i$  zu erzeugen, ist damit

$$\mathbb{P}_{\theta_2}(x_i) = \prod_{k=1}^L f_{x_{ik},0}. \quad (3.2)$$

Damit ergibt sich bei bekannten Parametern  $\theta = (\theta_1, \theta_2)$  und  $\lambda = (\lambda_1, \lambda_2)$  die Gesamtwahrscheinlichkeit für die Eingabe  $x$  wie folgt:

$$\begin{aligned} \mathbb{P}_{\theta,\lambda}(x) &= \prod_{i=1}^n \mathbb{P}_{\theta,\lambda}(x_i) \\ &= \prod_{i=1}^n \sum_{j=1}^2 \mathbb{P}_{\theta,\lambda}(x_i, Z_{ij} = 1) \\ &= \prod_{i=1}^n \sum_{j=1}^2 \mathbb{P}_{\theta,\lambda}(x_i \mid Z_{ij} = 1) \cdot \mathbb{P}_{\theta,\lambda}(Z_{ij} = 1) \\ &= \prod_{i=1}^n \sum_{j=1}^2 \mathbb{P}_{\theta_j}(x_i) \cdot \lambda_j. \end{aligned}$$

Hinweise: Das äußere Produkt zeigt lediglich, dass die  $x_i$  als unabhängig betrachtet werden. Die Summe zeigt, dass die Gesamtwahrscheinlichkeit für ein  $L$ -mer  $x_i$  sich aus zwei disjunkten Ereignissen zusammensetzt, nämlich “ $x_i$  aus Motivmodell” und “ $x_i$  aus Hintergrundmodell”.

Die likelihood-Funktion ist genau diese Wahrscheinlichkeit als Funktion der Parameter  $(\theta, \lambda)$ ; wir schreiben

$$L_x(\theta, \lambda) := \mathbb{P}_{\theta,\lambda}(x).$$

Ziel der Motivsuche ist, die Parameter so zu bestimmen (bei gegebenem  $x$ ), dass die likelihood maximal wird. Statt die likelihood zu maximieren, maximieren wir wie üblich ihren Logarithmus (log-likelihood); dies ist äquivalent, da der Logarithmus eine streng monoton wachsende Funktion ist.

$$\mathcal{L}_x(\theta, \lambda) := \log L_x(\theta, \lambda) = \sum_{i=1}^n \log \left[ \sum_{j=1}^2 \lambda_j \mathbb{P}_{\theta_j}(x_i) \right]. \quad (3.3)$$

Wie man erahnen kann, ist eine direkte Maximierung dieses Ausdrucks in allen Parametern schwierig. Hinzu kommt, dass die log-likelihood-Funktion aufgrund der Problemstellung höchstwahrscheinlich mehrere lokale Maxima aufweist: Es ist ja plausibel, dass mehrere Motive in der Eingabe versteckt sind.

Es ist natürlich möglich, beliebige nichtlineare Optimierungsroutinen für  $\mathcal{L}_x(\theta, \lambda)$  zu verwenden; dabei sollte jedoch bedacht werden, dass bei der DNA-Motivsuche  $3(L + 1) + 1$  freie Parameter zu bestimmen sind ( $3L$  aus  $\theta_1$ ,  $3$  aus  $\theta_2$ , einer aus  $\lambda$ ), also ein hochdimensionales Problem vorliegt. Statt die Optimierung in eine "black box" auszulagern, beschreiten wir einen anderen Weg.

Das Problem besteht darin, dass die Zugehörigkeiten  $Z_{ij}$  unbekannt sind. deswegen müssen wir, um  $\mathbb{P}_{\theta, \lambda}(x)$  zu erhalten, über alle möglichen Werte von  $Z_{ij}$  summieren; dadurch kommt in Gleichung (3.3) die Summe in den Logarithmus, und deswegen ist die Optimierung schwierig. Wenn wir aber so tun, als seien die  $Z_{ij}$  bekannt, wird alles wesentlich einfacher. Das Problem dabei ist, dass wir die Kenntnis von  $Z_{ij}$  irgendwo hernehmen müssen.

Wir behandeln im nächsten Abschnitt ganz allgemein den EM-Algorithmus, mit dem man die likelihood in probabilistischen Mehrkomponenten-Modellen (engl. *mixture models*) iterativ maximiert, indem man in einem Schritt die Zugehörigkeit  $Z_{ij}$  der  $x_i$  zu den einzelnen Komponenten schätzt und in einem anderen Schritt die Modellparameter  $(\theta, \lambda)$  für diese Schätzung so bestimmt, dass die likelihood dabei maximiert wird.

## 3.2 Der EM-Algorithmus für Mehrkomponentenmodelle

Wir betrachten folgende allgemeine Situation: Es gibt  $n$  Eingaben  $x_i$  der Dimension  $d$  über einem endlichen oder unendlichen Alphabet  $\Sigma$ , also  $x = (x_1, \dots, x_n)$  mit  $x_i = (x_{i1}, \dots, x_{id}) \in A^d$ . Dabei wird jedes  $x_i$  unabhängig von einem  $C$ -Komponenten-Modell erzeugt.

Zunächst wird Komponente  $j$  mit Wahrscheinlichkeit  $\lambda_j$  ausgewählt; dabei ist  $\lambda_j \geq 0$  für alle  $j$  und  $\sum_{j=1}^C \lambda_j = 1$ . Die gewählte Zugehörigkeit wird durch einen  $C$ -dimensionalen Vektor  $Z_i = (Z_{i1}, \dots, Z_{iC})$  ausgedrückt, so dass  $Z_{ij} = 1$  genau dann wenn  $x_i$  zu Komponente  $j$  gehört, und  $Z_{ij} = 0$  sonst.

Dann wird  $x_i$  nach einer für Komponente  $j$  spezifischen Verteilung gezogen, die wir mit  $\mathbb{P}_{\theta_j}$  bezeichnen;  $\theta_j$  beinhaltet dabei alle Parameter für Komponente  $j$ . Wir gehen davon aus, dass alle diese Modelle bekannt sind und damit  $\mathbb{P}_{\theta_j}(x_i)$  berechnet werden kann, wenn Parameter  $\theta_j$  bekannt sind.

Wir berechnen die Wahrscheinlichkeit, Komponente  $j$  auszuwählen und ein Datum  $x$  zu erzeugen:

$$\mathbb{P}_{\theta, \lambda}(x_i, Z_{ij} = 1) = \lambda_j \cdot \mathbb{P}_{\theta_j}(x_i).$$

Für einen beliebigen  $C$ -dimensionalen Einheitsvektor  $Z_i = (Z_{ij})$  schreiben wir dies als Produkt, bei dem wir über den Exponenten (0 oder 1) jeweils eine Komponente auswählen:

$$\mathbb{P}_{\theta, \lambda}(x_i, Z_i) = \prod_{j=1}^C [\lambda_j \cdot \mathbb{P}_{\theta_j}(x_i)]^{Z_{ij}}.$$

Damit ergibt sich die log-likelihood-Funktion zu allen Daten  $(x, Z)$  als

$$\mathcal{L}_{x,Z}(\theta, \lambda) = \sum_{i=1}^n \sum_{j=1}^C Z_{ij} \cdot [\log \lambda_j + \log \mathbb{P}_{\theta_j}(x_i)]. \quad (3.4)$$

Gegenüber der Gesamt-log-likelihood für die Daten  $x$  in (3.3) fällt die einfachere Struktur auf, aber eben auch, dass die unbekanntenen  $Z_{ij}$  auftreten.

Der EM-Algorithmus besteht nun darin, iterativ bessere Parameter für  $(\theta, \lambda)$  zu berechnen, indem

- sinnvolle oder zufällige Startwerte  $(\theta^0, \lambda^0)$  für die Parameter  $(\theta, \lambda)$  gewählt werden,
- im **E-Schritt** die aktuell zu optimierende Funktion  $f_{\theta^0, \lambda^0, x}(\theta, \lambda)$  als *Erwartungswert* der log-likelihood  $\mathcal{L}_{x,Z}(\theta, \lambda)$  gewählt wird; der Erwartungswert wird über die  $Z_{ij}$  gebildet; dabei wird die bedingte Verteilung der  $Z_{ij}$  gegeben  $x_i$  und die aktuellen Parameterwerte  $(\theta^0, \lambda^0)$  zugrunde gelegt; es ist also

$$f_{\theta^0, \lambda^0, x}(\theta, \lambda) := \mathbb{E}_{Z \mid (x; \theta^0, \lambda^0)} [\mathcal{L}_{x,Z}(\theta, \lambda)]. \quad (3.5)$$

- im **M-Schritt** (*Maximierung*) neue Parameterwerte  $(\theta^*, \lambda^*) = \operatorname{argmax}_{(\theta, \lambda)} f_{\theta^0, \lambda^0, x}(\theta, \lambda)$  berechnet werden und diese als neue Parameterwerte für die nächste Iteration verwendet werden.

Wir betrachten nun den E-Schritt genauer. Die erwartete log-likelihood über alle Wahlen von  $Z$  zu berechnen bietet sich an, da  $Z_{ij}$  in der Zielfunktion nur linear auftritt und der Erwartungswert ein linearer Operator ist. Es ergibt sich daher

$$\begin{aligned} f_{\theta^0, \lambda^0, x}(\theta, \lambda) &= \mathbb{E}_{Z \mid (x; \theta^0, \lambda^0)} [\mathcal{L}_{x,Z}(\theta, \lambda)] \\ &= \mathbb{E}_{Z \mid (x; \theta^0, \lambda^0)} \left[ \sum_{i=1}^n \sum_{j=1}^C Z_{ij} \cdot (\log \lambda_j + \log \mathbb{P}_{\theta_j}(x_i)) \right] \\ &= \sum_{i=1}^n \sum_{j=1}^C \mathbb{E}_{\theta^0, \lambda^0} [Z_{ij} \mid x_i] \cdot (\log \lambda_j + \log \mathbb{P}_{\theta_j}(x_i)) \\ &= \sum_{i=1}^n \sum_{j=1}^C Z_{ij}^0 \cdot (\log \lambda_j + \log \mathbb{P}_{\theta_j}(x_i)), \end{aligned}$$

wobei wir zur Abkürzung  $Z_{ij}^0 := \mathbb{E}_{\theta^0, \lambda^0} [Z_{ij} \mid x_i]$  gesetzt haben; diesen Wert rechnen wir jetzt aus. Da  $Z_{ij}$  eine Indikatorvariable ist, gilt

$$\begin{aligned} Z_{ij}^0 &= \mathbb{E}_{\theta^0, \lambda^0} [Z_{ij} \mid x_i] \\ &= \mathbb{P}_{\theta^0, \lambda^0}(Z_{ij} = 1 \mid x_i) \\ &= \frac{\mathbb{P}_{\theta^0, \lambda^0}(x_i \mid Z_{ij} = 1) \cdot \mathbb{P}_{\theta^0, \lambda^0}(Z_{ij} = 1)}{\mathbb{P}_{\theta^0, \lambda^0}(x_i)} \\ &= \frac{\lambda_j^0 \cdot \mathbb{P}_{\theta_j^0}(x_i)}{\sum_{k=1}^C \lambda_k^0 \cdot \mathbb{P}_{\theta_k^0}(x_i)}, \end{aligned} \quad (3.6)$$

wobei wir den Satz von Bayes für bedingte Wahrscheinlichkeiten angewendet haben; in der letzten Zeile wird wiederum die Wahrscheinlichkeit für  $x$  über alle Komponenten gewichtet summiert; es ist ja  $L_{x_i}(\theta^0, \lambda^0) = \mathbb{P}_{\theta^0, \lambda^0}(x_i) = \sum_{k=1}^C \lambda_k^0 \cdot \mathbb{P}_{\theta_k^0}(x_i)$  die Gesamt-likelihood von  $(\theta^0, \lambda^0)$  für das Datum  $x_i$ .

Die "erwartete Zugehörigkeit"  $Z_{ij}^0$  von  $x_i$  zu Komponente  $j$  ergibt sich also als Anteil der gewichteten Wahrscheinlichkeit von  $x_i$  aus Komponente  $j$  in Bezug auf die Gesamtwahrscheinlichkeit von  $x_i$  unter den Parametern  $(\theta^0, \lambda^0)$  und kann explizit ausgerechnet werden. Wir können damit auch die aktuelle erwartete log-likelihood  $f^0 := f_{\theta^0, \lambda^0, x}(\theta^0, \lambda^0)$  bestimmen.

Im M-Schritt werden die Parameter  $(\theta^*, \lambda^*)$  gesucht, die  $f_{\theta^0, \lambda^0, x}(\theta, \lambda)$  maximieren. Die Lösung für  $\lambda^*$  lässt sich allgemein angeben. Durch Einführen eines Lagrange'schen Multiplikators zur Wahrung der Bedingung  $\sum_{j=1}^C \lambda_j = 1$ , sowie Bilden und Nullsetzen der Ableitung  $\partial f / \partial \lambda_j$  erhält man als Maximierer

$$\lambda_j^* = \frac{1}{n} \sum_{i=1}^n Z_{ij}^0,$$

also die (intuitiv sinnvolle) durchschnittliche Zugehörigkeit aller Datenpunkte zu Komponente  $j$ .

Die Bestimmung von  $\theta^*$  ist naturgemäß modellabhängig; allgemein lässt sich immerhin die Ableitung nach dem  $\ell$ -ten Parameter  $\theta_{j\ell}$  der  $j$ -ten Komponente an der Stelle  $(\theta, \lambda)$  berechnen als

$$\frac{\partial f_{\theta^0, \lambda^0, x}}{\partial \theta_{j\ell}}(\theta, \lambda) = \sum_{i=1}^n Z_{ij}^0 \cdot \frac{\partial [\log \mathbb{P}_{\theta_j}(x_i)]}{\partial \theta_{j\ell}},$$

diese hängt nicht mehr von  $\lambda$  ab. Zur Berechnung von  $\theta^*$  müssen wir für das konkrete Modell die Ableitungen nach  $\theta_{j\ell}$  berechnen und Null setzen, gegebenenfalls unter Hinzunahme von Lagrange'schen Multiplikatoren bei Nebenbedingungen. Wir bringen an dieser Stelle ein einfaches Beispiel.

**3.2 Beispiel** (Gezinkter Würfel). Wir vermuten, dass ein betrügerischer Spieler mit zwei Würfeln arbeitet, einem fairen und einem gezinkten, aber wir wissen nicht, auf welche Weise der gezinkte Würfel gezinkt ist. In einer Runde wird jeweils  $d = 20$  mal mit demselben Würfel gewürfelt. Wir beobachten  $n = 1000$  Runden und wissen nicht, in welcher Runde welcher Würfel verwendet wurde. Wir wollen die Parameter (Augenwahrscheinlichkeiten) des gezinkten Würfels schätzen.

Offenbar ist  $C = 2$  (fairer Würfel und gezinkter Würfel); die Parameter für die erste Komponente stehen sogar mit  $\theta_1 = (1/6, \dots, 1/6)$  schon fest und müssen nicht geschätzt werden. Unbekannt sind die Wahrscheinlichkeiten  $\lambda_1$  und  $\lambda_2 = 1 - \lambda_1$ , mit denen jeweils der faire bzw. gezinkte Würfel gewählt wird und natürlich die Parameter  $\theta_2 = (\theta_{2,1}, \dots, \theta_{2,6})$  des gezinkten Würfels. Beobachtet werden die Augenzahlen der Würfe  $(x_{ik})$ ; nicht beobachtet werden kann die Würfelwahl  $Z_i$  für die  $i$ -te Runde (hier steht  $Z_i = (1, 0)$  für den fairen und  $Z_i = (0, 1)$  für den gezinkten Würfel).

Um den EM-Algorithmus zur Parameterschätzung anwenden zu können, müssen wir die Modelle für die beiden Komponenten angeben. Offensichtlich ist für den fairen Würfel

$$\mathbb{P}_{\theta_1}(x_i) = (1/6)^d = (1/6)^{20} \quad \text{für alle } 1 \leq i \leq n$$

eine Konstante. Für den gezinkten Würfel gilt

$$\mathbb{P}_{\theta_2}(x_i) = \prod_{k=1}^d \theta_{2,x_{ik}} = \prod_{q=1}^6 \theta_{2,q}^{y_{i,q}}, \quad \text{und}$$

$$\log \mathbb{P}_{\theta_2}(x_i) = \sum_{q=1}^6 y_{i,q} \cdot \log \theta_{2,q},$$

wobei  $y_{i,q} := \#\{k : x_{ik} = q\}$  gesetzt wurde (Häufigkeit von Augenzahl  $q$  in Versuch  $i$ ). Jetzt berechnen wir

$$\frac{\partial[\log \mathbb{P}_{\theta_2}(x_i)]}{\partial \theta_{2,q}} = \frac{y_{i,q}}{\theta_{2,q}}.$$

Für den M-Schritt ist die Nebenbedingung  $\sum_q \theta_{2,q} = 1$  zu beachten. Die Lagrange-Funktion

$$g(\theta, \lambda, \mu) := f_{\theta^0, \lambda^0, x}(\theta, \lambda) + \mu \cdot \left(1 - \sum_{q=1}^6 \theta_{2,q}\right)$$

hat die Ableitungen

$$\frac{\partial g}{\partial \theta_{2,q}}(\theta, \lambda, \mu) = -\mu + \sum_{i=1}^n Z_{i,2}^0 \cdot \frac{y_{i,q}}{\theta_{2,q}} \stackrel{!}{=} 0$$

$$\frac{\partial g}{\partial \mu}(\theta, \lambda, \mu) = 1 - \sum_{q=1}^6 \theta_{2,q} \stackrel{!}{=} 0$$

Dies ergibt die Lösung  $\theta_{2,q}^* = \frac{\sum_{i=1}^n Z_{i,2}^0 y_{i,q}}{\mu}$ , wobei  $\mu$  so gewählt werden muss, dass sich die  $\theta_{2,q}$  zu 1 summieren; also  $\mu = \sum_{q=1}^6 \sum_{i=1}^n Z_{i,2}^0 y_{i,q} = \sum_{i=1}^n Z_{i,2}^0 \sum_{q=1}^6 y_{i,q} = d \cdot \sum_{i=1}^n Z_{i,2}^0$ . Das führt auf das Endergebnis

$$\theta_{2,q}^* = \frac{\sum_{i=1}^n Z_{i,2}^0 \cdot y_{i,q} / d}{\sum_{i=1}^n Z_{i,2}^0}, \quad (3.7)$$

das intuitiv Sinn macht: Die Schätzung für  $\theta_{2,q}^*$ , also für die Wahrscheinlichkeit, mit dem gezinkten Würfel die Augenzahl  $q$  zu würfeln, ist der relative Anteil der Beobachtungen von  $q$  Augen, die Komponente 2 zugeordnet werden, in Bezug auf alle Beobachtungen, die Komponente 2 zugeordnet werden. Damit lässt sich jetzt ein EM-Schritt implementieren. Man überlege sich, inwiefern das Problem schwieriger wird, wenn man in jeder Runde nur einmal werfen darf ( $d = 1$ ). ♡

**3.3 Beispiel** (Mischung von Gaussverteilungen). Die Daten sind  $d$ -dimensionale Punkte  $x_i = (x_{i1}, \dots, x_{id})$ , die von  $C$  verschiedenen  $d$ -dimensionalen Normalverteilungen (Gaussverteilungen) erzeugt werden. Die  $j$ -te Normalverteilung wird mit Wahrscheinlichkeit  $\lambda_j$  ausgewählt, einen Punkt  $x_i$  zu generieren. Ihr Mittelpunkt sei  $\mu_j \in \mathbb{R}^d$ ; ihre Kovarianzmatrix sei  $\sigma_j^2 \cdot \text{Id}$  (ein skalares Vielfaches der Einheitsmatrix) mit  $\sigma_j > 0$ , so dass die Niveaulinien konzentrische Kreise um  $\mu_j$  bilden. Damit beinhaltet  $\theta_j = (\mu_j, \sigma_j)$  die Parameter der  $j$ -ten Komponente, und die Dichte ergibt sich als

$$p_{\theta_j}(x_i) = \frac{1}{(2\pi\sigma_j^2)^{d/2}} \cdot \exp\left(-\frac{\|x_i - \mu_j\|^2}{2\sigma_j^2}\right).$$

Obwohl wir jetzt nicht über Wahrscheinlichkeiten, sondern über Dichten sprechen, bleiben die gemachten Aussagen gültig, wenn wir  $\mathbb{P}$  (für Wahrscheinlichkeiten) wo nötig durch  $p$  (für Dichten) ersetzen.

Zum Ausführen des M-Schritts berechnen wir  $\frac{\partial[\log p_{\theta_j}(x_i)]}{\partial \mu_j}$  und  $\frac{\partial[\log p_{\theta_j}(x_i)]}{\partial \sigma_j^2}$ , setzen die Ableitungen gleich Null und erhalten die intuitiv einleuchtenden Ergebnisse

$$\begin{aligned}\mu_j^* &= \frac{\sum_{i=1}^n Z_{ij}^0 x_i}{\sum_{i=1}^n Z_{ij}^0}, \\ (\sigma_j^2)^* &= \frac{\sum_{i=1}^n Z_{ij}^0 \|x_i - \mu_j^*\|^2 / d}{\sum_{i=1}^n Z_{ij}^0}.\end{aligned}$$

Zur Übung wird empfohlen, dies nachzurechnen. ♡

Wir sehen, dass eine Iteration des EM-Algorithmus relativ einfach durchzuführen ist, sofern Implementierungen der

1. komponentenspezifischen Wahrscheinlichkeitsfunktionen
2. Maximum-likelihood-Schätzer der Parameter

zur Verfügung stehen. Dabei werden aus den “alten” Parametern  $(\theta^0, \lambda^0)$  “neue” Parameter  $(\theta^*, \lambda^*)$  berechnet, indem eine mit Hilfe der alten Parameter berechnete erwartete log-likelihood (3.5) maximiert wird. Die Frage, die wir bisher nicht beantwortet haben, ist: Hilft dies auch, unser eigentliches Ziel zu erreichen, nämlich die Gesamt-likelihood

$$\mathcal{L}_x(\theta, \lambda) := \log L_x(\theta, \lambda) = \sum_{i=1}^n \log \left[ \sum_{j=1}^C \lambda_j \mathbb{P}_{\theta_j}(x_i) \right], \quad (3.8)$$

vgl. (3.3), zu maximieren? Die positive Antwort gibt der folgende Satz.

**3.4 Satz.** *Nach einem EM-Schritt mit  $(\theta^0, \lambda^0) \mapsto (\theta^*, \lambda^*)$  gilt*

$$\mathcal{L}_x(\theta^*, \lambda^*) \geq \mathcal{L}_x(\theta^0, \lambda^0);$$

*die Gesamt-likelihood verbessert sich also in jedem Schritt.*

**Beweis.** Zu Abkürzung schreiben wir  $\phi := (\theta, \lambda)$  und definieren ferner für feste Daten  $x$  und gegebene Parameter  $\phi^0$  die Funktion

$$Q(\phi) := f_{\theta^0, \lambda^0, x}(\theta, \lambda) = \mathbb{E}_{Z \mid (x, \phi^0)} [\mathcal{L}_{x, Z}(\phi)];$$

diese Funktion wird im M-Schritt des EM-Algorithmus in  $\phi$  maximiert und liefert  $\phi^*$ . Deswegen gilt  $Q(\phi^*) \geq Q(\phi^0)$ . Genauer ist

$$\begin{aligned}0 \leq Q(\phi^*) - Q(\phi^0) &= \sum_{i=1}^n \sum_{j=1}^C Z_{ij}^0 \log \frac{\lambda_j^* \mathbb{P}_{\theta_j^*}(x_i)}{\lambda_j^0 \mathbb{P}_{\theta_j^0}(x_i)} \\ &= \sum_{i=1}^n \sum_{j=1}^C Z_{ij}^0 \log \frac{\lambda_j^* \mathbb{P}_{\theta_j^*}(x_i)}{Z_{ij}^0 \mathbb{P}_{\phi^0}(x_i)},\end{aligned}$$

### 3 Motivsuche mit dem EM-Algorithmus

da nach (3.6) genau  $Z_{ij}^0 \mathbb{P}_{\phi^0}(x_i) = \lambda_j^0 \mathbb{P}_{\theta_j^0}(x_i)$ . Wir zeigen jetzt, dass die log-likelihood in jedem EM-Schritt um mindestens diese nichtnegative Differenz zunimmt.

Wir betrachten jetzt die log-likelihood-Differenz

$$\begin{aligned}
 \mathcal{L}_x(\phi^*) - \mathcal{L}_x(\phi^0) &= \sum_{i=1}^n \log \frac{\mathbb{P}_{\phi^*}(x_i)}{\mathbb{P}_{\phi^0}(x_i)} \\
 &= \sum_{i=1}^n \log \frac{\sum_{j=1}^C \lambda_j^* \mathbb{P}_{\theta_j^*}(x_i) \cdot \frac{Z_{ij}^0}{Z_{ij}^0}}{\mathbb{P}_{\phi^0}(x_i)} \\
 &= \sum_{i=1}^n \log \sum_{j=1}^C Z_{ij}^0 \frac{\lambda_j^* \mathbb{P}_{\theta_j^*}(x_i)}{Z_{ij}^0 \mathbb{P}_{\phi^0}(x_i)} \\
 &\geq \sum_{i=1}^n \sum_{j=1}^C Z_{ij}^0 \log \frac{\lambda_j^* \mathbb{P}_{\theta_j^*}(x_i)}{Z_{ij}^0 \mathbb{P}_{\phi^0}(x_i)} \\
 &= Q(\phi^*) - Q(\phi^0).
 \end{aligned}$$

Die  $\geq$ -Abschätzung ist die Jensen'sche Ungleichung  $\log \sum_j \mu_j v_j \geq \sum_j \mu_j \log v_j$ , die für beliebige positive  $v_j$  und Wahrscheinlichkeitsverteilungen  $\mu = (\mu_j)$  wie z.B.  $(Z_{ij}^0)$  direkt aus der Konkavität der log-Funktion folgt ("der Logarithmus einer Konvexkombination ist größer gleich der Konvexkombination der Logarithmen").  $\square$

### 3.3 Motivsuche mit dem EM-Algorithmus

Nach der allgemeinen Diskussion des EM-Algorithmus für Mehrkomponentenmodelle (und Beispielen, die nichts mit Motivsuche zu tun haben), kehren wir zum Thema Motivsuche zurück. Wir wissen nun, dass sich die Parameter  $\theta_1 = (f_{a,k})$  der PFM für  $a \in A$  und  $1 \leq k \leq L$  und die Parameter  $\theta_2 = (f_{a,0})$  des Hintergrundmodells ebenso wie die relativen Modellhäufigkeiten  $\lambda_1, \lambda_2$  mit dem EM-Algorithmus iterativ schätzen lassen. Ungeklärt dabei sind noch folgende Fragen, die wir jetzt angehen:

- Worin besteht konkret der M-Schritt?
- Wie werden sinnvolle Startparameter  $(\theta_1, \theta_2, \lambda)$  gewählt?
- Wie gehen wir mit der Tatsache um, dass im ursprünglichen Problem Sequenzen und keine unabhängigen  $L$ -Tupel gegeben sind; welche Auswirkungen hat die Umformulierung in  $L$ -Tupel auf die Ergebnisse?
- Wie kann man mehrere verschiedene Motive finden, die nicht mit bereits gefundenen überlappen?

**Ausführung des M-Schritts.** Wir müssen für PFM- und Hintergrundmodell noch die ML-Schätzer der neuen Parameter berechnen. Dazu bemerken wir, dass die gleiche Situation vorliegt wie im Beispiel des gezinkten Würfels (Beispiel 3.2). Sei  $y_{aik} := 1$ , wenn  $x_{ik} = a \in A$ ;

sei  $y_{aik} := 0$  sonst. Sei ferner  $y_{ai0} := \sum_{k=1}^L y_{aik}$  die Anzahl der  $a$ -Symbole in  $x_i$ . Damit lässt sich leicht ausrechnen, dass

$$f_{a,k}^* = \frac{\sum_{i=1}^n Z_{i1}^0 y_{aik}}{\sum_{i=1}^n Z_{i1}^0} \quad \text{in der PFM, und}$$

$$f_{a,0}^* = \frac{\sum_{i=1}^n Z_{i2}^0 y_{ai0}/L}{\sum_{i=1}^n Z_{i2}^0} \quad \text{im Hintergrund.}$$

**Wahl der Startparameter.** Die Parameter  $\theta_2 = (f_{a,0})$  des Hintergrundmodells bereiten keine Probleme; hier nimmt man die relativen Symbolhäufigkeiten gemittelt über allen  $L$ -Tupel  $x_i$ . Ebenso fällt die ad-hoc Wahl von beispielsweise  $\lambda = (0.01, 0.99)$  nicht schwer: Der Anteil der Motive an der Gesamtsequenz sollte relativ gering sein. Es bleibt, entweder geeignete  $\theta_1 = (f_{a,k})$  für Positionen  $1 \leq k \leq L$  und Symbole  $a \in A$  auszuwählen, oder aber Indizes  $i$  zu wählen, für die  $Z_{i1} = 1$  sein soll (also unter den  $x_i$  Motivinstanzen zu wählen).

**Korrektur für überlappende Motive.**

**Mehrere verschiedene Motive.**



---

## Identifizierbarkeit in Signal-Sensor-Netzwerken

---

In diesem Kapitel befassen wir uns (abstrakt) mit Signal-Sensor-Netzwerken und gehen der Frage nach, unter welchen Bedingungen man quantitative Konnektivitätsinformationen und Signalstärken mit Hilfe von Messungen ermitteln kann, die Linearkombinationen der eigentlichen Signale sind. Diese Fragen führen auf eine reiche Theorie mit Verbindungen zwischen linearer Algebra und diskreter Mathematik. Der Inhalt dieses Kapitels basiert zu großen Teilen auf der Dissertation von Epameinondas Fritzilas ?.

### 4.1 Motivation

Wir bringen zur Abwechslung ein nichtbiologisches Beispiel für ein Signal-Sensor-Netzwerk.

**Die abgehörte Cocktailparty.** Auf einer Cocktailparty stehen mehrere Personen in verschiedenen Gruppen zusammen und unterhalten sich. Der Gastgeber möchte die Gespräche (aus welchen Gründen auch immer) aufnehmen und hat dazu an verschiedenen Stellen im Raum Mikrofone versteckt. Natürlich nimmt jedes Mikrofon eine Mischung aller Konversationen auf, und zwar um so besser, je näher das Mikrofon der Gruppe ist. Ab einer gewissen Distanz wird das Signal nicht mehr aufgenommen bzw. geht im Rauschen unter.

Formal ist eine Menge  $C$  von Signalquellen (Gespräche) und eine Menge  $R$  von Sensoren (Mikrofone) gegeben. Jede Sensormessung  $y_i$  ( $i \in R$ ) ist eine "Mischung", also Linearkombination, der Signale  $x_j$ ; die Mischungskoeffizienten  $a_{ij}$  ergeben sich in diesem Beispiel physikalisch aufgrund des Abstands zwischen Sensor  $i$  und Signal  $j$ , aber wir nehmen im Allgemeinen an, dass die Abstände und damit die Koeffizienten nicht genau bekannt sind (der Gastgeber

hat vergessen, eine Videoüberwachung zu installieren); es sei lediglich offensichtlich welche Koeffizienten bestimmt Null sind (sehr großer Abstand). Das Modell lautet also (zunächst)

$$y_i = \sum_{j \in C} a_{ij} x_j, \quad \text{oder}$$

$$y = A \cdot x$$

mit  $y \in \mathbb{R}^m$ ,  $A \in \mathbb{R}^{m \times n}$  und  $x \in \mathbb{R}^n$ , wobei  $m = |R|$  und  $n = |C|$ .

Hierbei haben wir Rauschen, Messungenauigkeiten, etc. vernachlässigt. In der Praxis wird das Gleichungssystem niemals exakt erfüllt sein. Durch Einführen eines Fehlervektors  $e$  erhält man  $y = Ax + e$  oder  $e = y - Ax$  und kann nun versuchen, den Fehler (in irgendeiner Norm, üblicherweise der Euklidischen Norm) zu minimieren.

**Lineare Ausgleichsprobleme bei bekannter Koeffizientenmatrix.** Ganz abstrakt ist häufig das folgende Problem zu lösen: Zu gegebenem  $A$  und  $y$  und zu einer gegebenen konvexen Fehlerfunktion  $f : \mathbb{R}^m \rightarrow \mathbb{R}$ , einer konvexen Regularisierungsfunktion  $r : \mathbb{R}^n \rightarrow \mathbb{R}$  und einem Gewichtungsfaktor  $c$ , finde  $x \in \mathbb{R}^n$ , so dass  $f(y - Ax) + c \cdot r(x)$  minimiert wird. Beachte:  $f$  ist auf dem Mess- oder Sensorraum  $\mathbb{R}^m$  definiert, aber  $r$  auf dem Signalraum  $\mathbb{R}^n$ . Man will also einerseits durch die Wahl von  $x$  die Differenz  $y - Ax$  minimieren, andererseits auch bestimmte  $x$  gegenüber anderen bevorzugen (zum Beispiel solche mit kleinen Einträgen); die relative Gewichtung dieser beiden Ziele stellt man über den Faktor  $c$  ein.

Wir erwähnen ein Beispiel, bei dem  $f$  die euklidische Länge ist (bzw. die Summe quadratischer Fehler) und  $r$  nicht vorkommt. Allerdings sollen unter allen äquivalenten Lösungen (wenn es mehrere gibt) diejenige mit kleinster Länge ausgewählt werden.

Also: Ist die Koeffizientenmatrix  $A$  bekannt, wird  $y$  gemessen und verwendet man die (quadratische) Euklidische Norm, so erhält man das klassische Problem "Zu gegebenem  $(A, y)$  finde  $x$ , so dass  $\|Ax - y\|_2^2$  (Summe der Fehlerquadrate) minimal wird."

In der Praxis wird ein solches lineares Ausgleichsproblem mit Hilfe der Singulärwertzerlegung (Abschnitt A.4) und der Pseudoinversen (Abschnitt A.5) gelöst; mit der dortigen Notation für die Pseudoinverse  $A^\dagger$  ist das gesuchte  $x = A^\dagger y$ .

**Unbekannte Koeffizienten: Fragen zur Eindeutigkeit** Im Gegensatz zu den obigen Überlegungen wollen wir aber  $A$  nicht als bekannt voraussetzen! Uns interessiert auch gar nicht so sehr die tatsächliche Lösung des entsprechenden Problems, sondern zunächst grundsätzlich die Lösbarkeit und Eindeutigkeit der Lösung. Hieraus leiten wir dann ein Optimierungsproblem zur Auswahl optimaler Sensoren ab und befassen uns also mit dem *Netzwerkdesign*. Wir setzen voraus, dass konvexe Probleme, wie sie oben beschrieben wurden mit der richtigen Software schon irgendwie gelöst werden können. (Ich kann ferner den Besuch einer weiteren Spezialvorlesung "Konvexe Optimierung" empfehlen.)

Die Frage hier lautet also: Wie eindeutig ist folgendes Problem lösbar? Gegeben  $y$ , finde  $(A, x)$ , so dass ein Fehlermaß  $f(y - Ax)$  minimal ist. Intuitiv ahnt man, dass das nicht gutgehen kann: Es sind nur  $m$  Werte gegeben, aber  $mn + n$  Werte unbekannt.

Wir können die Situation verbessern, indem wir nicht nur eine Messung, sondern  $p$  unabhängige Messungen betrachten; die Koeffizienten  $a_{ij}$  bleiben dabei gleich. Wir haben jetzt

eine  $m \times p$ -Matrix  $Y = (y_{it})$  von Messwerten, eine  $m \times n$ -Matrix  $A = (a_{ij})$  von Koeffizienten und eine  $n \times p$ -Matrix  $X = (x_{jt})$  von Signalen.

Zu gegebenem  $Y$  suchen wir  $(A, X)$ ; es soll  $Y \approx AX$  gelten, also ein Fehlermaß  $f(Y - AX)$  mit einer konvexen Funktion  $f : \mathbb{R}^{m \times p} \rightarrow \mathbb{R}$  minimiert werden. Dies erscheint nicht mehr unmöglich: Gegeben sind  $mp$  Werte, gesucht sind  $mn + np$ . Damit das überhaupt(!) funktionieren kann, müssen wir genügend Experimente machen, so dass  $mp \geq mn + np$  gilt. Dies ist erfüllt, wenn  $m > n$  ist (mehr Sensoren als Signale) und  $p \geq mn/(m-n)$  gilt. Haben wir beispielsweise doppelt so viele Sensoren wie Signale, also  $m = 2n$ , dann benötigen wir  $p \geq 2n$  Experimente. Ist allgemeiner  $m = cn$  mit  $c > 1$ , benötigen wir mindestens  $c/(c-1) \cdot n$  Experimente. Dies garantiert aber noch nicht die eindeutige Lösbarkeit, sondern formuliert lediglich die notwendige Bedingung, dass wir nicht mehr Unbekannte bestimmen können als uns Daten vorliegen.

Eine andere Überlegung liefert sofort neue schlechte Nachrichten. Wenn wir eine Lösung  $(A^*, X^*)$  gefunden haben und  $M$  eine beliebige invertierbare  $n \times n$ -Matrix ist, dann ist offensichtlich

$$A^*X = A^*MM^{-1}X = A'X'$$

mit  $A' := A^*M$  und  $X' = M^{-1}X^*$ . Jedes invertierbare  $M$  liefert also eine neue äquivalente Lösung; von Eindeutigkeit kann keine Rede sein.

An dieser Stelle müssen wir einen Joker ziehen und zusätzliches Wissen einbringen; ansonsten wäre das Kapitel jetzt zu Ende. Die Situation verbessert sich offensichtlich, wenn wir etwas über  $A$  wissen, zum Beispiel, dass viele der Einträge (und auch welche der Einträge) Null sind. Die Aussage  $a_{ij} = 0$  bedeutet ja, dass Sensor  $i$  das Signal  $j$  gar nicht misst, bzw. dass der Anteil so unbedeutend ist, dass er im Fehlerterm untergeht. Wir befassen uns nun genau mit dieser Frage: Welche Aussagen können wir über die Eindeutigkeit einer Lösung  $(A^*, X^*)$  machen, wenn von  $A$  bekannt ist, dass an bestimmten Stellen Nullen stehen müssen?

**Beispiel: Microarrays.** Bevor wir das Problem formal betrachten, liefern wir noch ein relevantes Beispiel aus der Bioinformatik, das wir bereits kennen: DNA-Microarrays. Wir erinnern uns, dass hier einsträngige DNA-Sonden an Gen-Transkripte hybridisieren, die einen Teilstring enthalten, dessen Sequenz revers komplementär zur Sonde ist. Nun sollten die DNA-Sonden so gewählt sein, dass jeweils nur ein Gen, das sogenannte Zielgen, daran hybridisiert, aber erstens ist dies nicht immer möglich und zweitens funktioniert die Hybridisierung auch (wenn auch nicht ganz so stabil), wenn das Transkript nicht das perfekte reverse Komplement enthält, sondern nur eine Approximation davon (beispielsweise einen genügend langen gemeinsamen Teilstring).

Wir messen also in vielen Experimenten (indirekt) die Hybridisierungsstärke  $y_i$  der Sonden  $i \in R$ , die sich als gewichtete Summe der eigentlichen Expressionswerte  $x_j$  der Gene oder Transkripte  $j \in C$  ergibt; die Gewichte sind genau die  $a_{ij}$ . Besteht nun nahezu überhaupt keine Sequenzähnlichkeit zwischen Sonde  $i$  und dem reversen Komplement von Transkript  $j$ , darf man  $a_{ij} = 0$  voraussetzen, ansonsten ist dieser Wert unbekannt und soll aus den Daten bestimmt werden. Die Frage, die wir gestellt haben, lautet: Ist dies eindeutig möglich?

## 4.2 Identifizierbarkeit

Wir formalisieren nun das Problem, das wir im letzten Abschnitt aufgeworfen haben.

**4.1 Definition** (Nullmuster). Ein  $m \times n$ -Nullmuster ist eine Matrix  $Z = (z_{ij}) \in \{0, 1\}^{m \times n}$ . Eine Matrix  $A = (a_{ij}) \in \mathbb{R}^{m \times n}$  gehorcht einem Nullmuster  $Z$ , geschrieben  $A \triangleleft Z$ , wenn aus  $z_{ij} = 0$  auch  $a_{ij} = 0$  folgt. (Aus  $z_{ij} = 1$  folgt nichts.)

Ein Nullmuster für  $A$  schreibt also vor, wo auf jeden Fall Nullen stehen; die anderen Einträge können “zufällig” Null sein, müssen es aber nicht.

Wir kommen zurück zu der Frage, wie man aus einer Lösung  $(A^*, X^*)$  eine andere  $(A', X')$  mit  $A^* X^* = A' X'$  bekommen kann? Wir hatten schon gesehen, dass man durch “Zwischenschalten” einer invertierbaren  $n \times n$ -Matrix  $M$  die Lösung  $A' = A^* M$ ,  $X' = M^{-1} X^*$  erhält.

In der Tat lassen sich *alle* Lösungen so erhalten, wenn man geeignete Voraussetzungen macht: Wir setzen  $n \leq \min\{m, p\}$  voraus; sonst macht das Problem keinen Sinn. Wir beschränken uns ferner auf Lösungen  $(A, X)$ , bei denen  $A$  und  $X$  den vollen Rang  $n$  haben.

**4.2 Lemma.** *Es seien  $A^*, A' \in \mathbb{R}^{m \times n}$  mit Rang  $n$ , sowie  $X^*, X' \in \mathbb{R}^{n \times p}$  mit Rang  $n$ . Ist  $A^* X^* = A' X'$ , dann gibt es eine invertierbare Matrix  $M \in \mathbb{R}^{n \times n}$  mit  $A' = A^* M$  und  $X' = M^{-1} X^*$ .*

Die interessante Frage ist nun, welche Einschränkungen an  $M$  ein gegebenes Nullmuster  $Z$  für  $A$  liefert. (Wir lassen nur solche Nullmuster zu, die nicht verhindern, dass  $A$  vollen Rang  $n$  hat.) Es muss nun sowohl  $A^* \triangleleft Z$  als auch  $A' = A^* M \triangleleft Z$  gelten.

Es ist  $a'_{ij} = \sum_k a_{ik} m_{kj}$ . Sei nun  $z_{ij} = 0$ ; dann muss also auch  $a'_{ij} = \sum_k a_{ik} m_{kj} = 0$  sein. Wir betrachten diese Bedingungen spaltenweise, also jetzt für ein fest gewähltes  $j$ . Dann lautet die Bedingung

$$\sum_k a_{ik} m_{kj} = 0 \quad \text{für alle } i \in I(j).$$

Wir schreiben dies in Matrixform. Sei  $I(j) := \{i \mid z_{ij} = 0\}$  die Indexmenge der Zeilen, die in der  $j$ -ten Spalte eine Null vorschreiben. Sei  $M_j := (m_{kj})$  die  $j$ -te Spalte von  $M$ . Sei  $[n] := \{1, \dots, n\}$ . Das Gleichungssystem lautet nun

$$A_{I(j), [n]} \cdot M_j = 0;$$

dies ist ein homogenes lineares Gleichungssystem mit  $|I(j)|$  Zeilen und  $n$  Spalten.

Wir bemerken, da ja auch  $A \triangleleft Z$  gilt, dass die  $j$ -te Spalte von  $A_{I(j), [n]}$  nach Konstruktion nur aus Nullen besteht. Diese beschreibt die Koeffizienten von  $m_{jj}$ , d.h., das  $j$ -te Diagonalelement geht nur mit Null-Koeffizienten (also gar nicht) in das System ein. Das verwundert nicht: Wählt man für  $M$  eine Diagonalmatrix  $D$ , dann haben  $A' = A^* D$  und  $X' = D^{-1} X^*$  an den gleichen Stellen Nullen. Für  $A'$  werden lediglich die Spalten von  $A^*$  umskaliert und für  $X'$  passend invers dazu die Zeilen von  $X^*$ . Wir bekommen also durch das Nullmuster keinerlei Informationen über die Diagonalelemente von  $M$ . Das ist allerdings nicht so schlimm, da es sich um eine reine Skalierungsfrage handelt und man diese häufig durch externe Information einfach festlegen kann (indem man beispielsweise die Einheiten der Messungen festlegt).

Was bleibt zu hoffen? Wir streichen die  $j$ -te Spalte von  $A_{I(j),[n]}$  sowie  $m_{jj}$  aus  $M_j$  und erhalten mit  $\tilde{A}_j := A_{I(j),[n]\setminus\{j\}}$  und  $\tilde{M}_j := (m_{1j}, \dots, m_{j-1,j}, m_{j+1,j}, \dots, m_{nj})^T$  das System

$$\tilde{A}_j \cdot \tilde{M}_j = 0.$$

Der Lösungsraum beschreibt die Möglichkeiten für die  $j$ -te Spalte der Matrix  $M$  (ohne Diagonalelement  $m_{jj}$ ).

Idealerweise hat dieses System nur die Lösung  $\tilde{M}_j = 0$ . Dies ist genau dann der Fall, wenn die Matrix  $\tilde{A}_j$  vollen Rang  $n - 1$  hat, also die verbleibenden  $n - 1$  Spalten linear unabhängig sind. Wir halten dies in einem Lemma fest.

**4.3 Lemma.** *Eine Lösung  $(A, X)$  von  $Y \approx AX$  ist genau dann bis auf Skalierung eindeutig, wenn alle Matrizen  $\tilde{A}_j = A_{I(j),[n]\setminus\{j\}}$  den Rang  $n - 1$  haben.*

Ärgerlich an Lemma 4.3 ist die Abhängigkeit des Resultats von den konkreten (unbekannten) Werten von  $A$ . Scheitert die Eindeutigkeit, kann das zwei Gründe haben: Einerseits kann das Nullmuster bereits verhindern, dass einige der Matrizen  $\tilde{A}_j$  den Rang  $n - 1$  erreichen können; andererseits kann das Nullmuster das (im Prinzip) zulassen, und wir haben lediglich Pech mit den konkreten Werten von  $A$ .

Wir wollen die Frage so stellen, dass die Antwort nur vom Nullmuster  $Z$  abhängt. Daher betrachten wir den maximal möglichen Rang, den eine Matrix  $A$ , die dem Nullmuster  $Z$  gehorcht, überhaupt haben kann, und nennen dies den strukturellen Rang von  $Z$ .

**4.4 Definition.** Der *strukturelle Rang* eines Nullmusters  $Z$  ist

$$\text{srang}(Z) := \max_{A \triangleleft Z} \text{rang}(A).$$

**4.5 Definition** (Identifizierbarkeit). Sei  $Z$  ein  $m \times n$ -Nullmuster mit  $m \geq n$ . Sei  $\tilde{Z}_j := Z_{I(j),[n]\setminus\{j\}}$  das Teilmuster, das entsteht, indem man die  $j$ -te Spalte und alle Zeilen  $i$  mit  $Z_{ij} = 1$  weglässt. Wir nennen Spalte  $j$  eines  $m \times n$ -Nullmusters  $Z$  *strukturell identifizierbar* oder einfach *identifizierbar*, wenn  $\text{srang}(\tilde{Z}_j) = n - 1$ . Wir nennen  $Z$  insgesamt identifizierbar, wenn jede Spalte identifizierbar ist.

Die Tatsache, dass Spalte  $j$  identifizierbar ist, bedeutet anschaulich, dass die  $j$ -te Spalte von  $A'$  einfach ein Vielfaches von der  $j$ -ten Spalte von  $A^*$  ist, also bis auf Skalierung eindeutig bestimmt ist.

Die wesentliche Frage ist nun, wie man den strukturellen Rang eines Nullmusters effizient berechnet. Es scheint nicht praktikabel, unendlich viele Matrizen, die dem Nullmuster gehorchen, zu betrachten und deren Ränge auszurechnen. Hier kommt uns ein nützlicher Zusammenhang zur diskreten Mathematik zur Hilfe.

Zunächst stellen wir fest, dass man ein Nullmuster  $Z$  kanonisch als bipartiten Graph darstellen kann, indem man Zeilen (Sensoren)  $R$  und Spalten  $C$  (Signale) von  $Z$  als die beiden Knotenmengen auffasst und Einsen in  $Z$  als Kanten.

**4.6 Definition** (Bipartiter Graph zu einem Nullmuster). Zu  $Z \in \{0, 1\}^{m \times n}$  sei  $R := \{R_i \mid i = 1, \dots, m\}$  (Sensorknoten) und  $C := \{C_j \mid j = 1, \dots, n\}$  (Signalknoten). Der zugehörige bipartite Graph  $G_Z = (V_Z, E_Z)$  ist definiert durch  $V_Z := R \cup C$  und  $E := \{\{R_i, C_j\} \mid Z_{ij} = 1\}$ .

Der Begriff der Identifizierbarkeit wird analog auf die  $C$ -Knoten von  $G_Z$  und  $G_Z$  insgesamt übertragen.

Es gilt nun folgende Aussage, die direkt einen effizienten Algorithmus zur Berechnung des strukturellen Rangs impliziert.

**4.7 Satz.** *Der strukturelle Rang  $\text{srang}(Z)$  ist identisch mit der Kardinalität eines größten Matchings in  $G_Z$ . (Ein Matching in einem Graphen  $G = (V, E)$  ist eine Kantenmenge  $M \subseteq E$ , die die Eigenschaft hat, dass jeder Knoten zu höchstens einer Kante in  $M$  inzident ist.)*

Bemerkung: Man muss unterscheiden zwischen maximalen Matchings (zu denen man keine Kante hinzufügen kann, ohne die Matching-Eigenschaft zu verletzen) und Matchings maximaler Kardinalität, also Matchings mit den meisten Kanten. Im Englischen spricht man von “maximal” und “maximum”; wir werden einfach “größtes” Matching für ein Matching maximaler Kardinalität, also ein maximum matching, sagen.

**Beweis.** Wie immer, wenn die Gleichheit zweier Größen zu beweisen ist, teilen wir den Beweis in zwei Teile,  $\leq$  und  $\geq$ .

“ $\geq$ ”: Sei  $M$  ein größtes Matching in  $G_Z$  mit  $|M| =: k$ . Interpretiere  $M$  als binäre Matrix  $A_M$ . Offensichtlich gilt  $A_M \triangleleft Z$ , da  $M$  nur Kanten verwendet, die in  $G_Z$  vorkommen. Nun enthält  $A_M$  wegen der Matching-Eigenschaft in genau  $k$  Zeilen und in genau  $k$  Spalten jeweils genau eine Eins, ansonsten nur Nullzeilen und Nullspalten. Damit hat  $A_M$  den Rang  $k$ , und es ist  $\max_{A \triangleleft Z} \text{rang}(A) = \text{srang}(Z) \geq k$ .

“ $\leq$ ”: Es sei  $\text{srang}(Z) = \max_{A \triangleleft Z} \text{rang}(A) = k$ ; dann gibt es also mindestens eine Matrix  $A^*$  mit  $\text{rang}(A^*) = k$ . Wir behaupten: Es gibt (mindestens)  $k$  paarweise verschiedene Zeilen  $i_1 < \dots < i_k$  und Spalten  $j_1 < \dots < j_k$ , so dass  $A_{i_q, j_q} \neq 0$  für alle  $q = 1, \dots, k$ . Gäbe es diese nicht, würde folgen, dass  $A^*$  mehr als  $n - k$  Nullzeilen oder Nullspalten hat, was einen Widerspruch zum Rang  $k$  wäre. Also existieren in  $G_Z$  die Kanten  $\{\{R_{i_q}, C_{j_q}\} \mid q = 1, \dots, k\}$ . Diese Kantenmenge ist aber auch ein Matching, also haben wir ein Matching in  $G_Z$  der Kardinalität  $k$  gefunden. Also hat ein größtes Matching eine Kardinalität  $\geq k$ . Damit ist  $k = \text{srang}(Z)$  kleiner gleich der Kardinalität eines größten Matchings.  $\square$

Man wird also folgendermaßen vorgehen: Zu gegebenem  $Z$ , bilde zunächst den Graphen  $G_Z$ . Iteriere folgende Schritte über alle Spalten  $j$ : Bilde den Teilgraphen  $G_{\bar{Z}_j}$ , indem  $C_j$  und seine adjazenten Knoten in  $R$ , sowie alle dazu inzidenten Kanten aus  $G_Z$  entfernt weredn. Finde ein größtes Matching in  $G_{\bar{Z}_j}$ . Hat dies die Kardinalität  $n - 1$ , ist also jeder  $C$ -Knoten gematcht, dann ist Spalte  $j$  identifizierbar, ansonsten nicht.

**Matchings maximaler Kardinalität in bipartiten Graphen.** Größte Matchings in bipartiten Graphen  $G = (V = (R \cup C), E)$  lassen sich effizient finden; hierzu gibt es verschiedene Algorithmen. Alle basieren darauf, zu einem existierenden Matching  $M$ , das leer sein kann, einen (oder mehrere)  $M$ -erweiternde(n) Pfad(e) zu finden. Ein  $M$ -erweiternder Pfad besteht alternierend aus Kanten, die zu  $M$  gehören, und solchen, die nicht zu  $M$  gehören. An den Endpunkten liegt jeweils ein freier Knoten in  $C$  und ein freier Knoten in  $R$ . Frei heißt hier, dass keine Kante aus  $M$  inzident ist. Vertauscht man in so einem Pfad  $M$ -Kanten mit nicht  $M$ -Kanten, erhält man ein um eins größeres Matching. (Ein  $M$ -erweiternder Pfad muss keine  $M$ -Kanten benutzen; der einfachste und schönste Fall ist der, dass ein freier  $R$ -Knoten direkt mit einem freien  $C$ -Knoten durch eine Kante verbunden ist; diese Kante ist ein  $M$ -erweiternder Pfad; man vergrößert  $M$  einfach dadurch, dass man sie hinzunimmt.) Ein Satz sagt, dass  $M$  genau dann maximal ist, wenn kein  $M$ -erweiternder Pfad existiert. Der entsprechende Algorithmus hat Komplexität  $O(|V||E|)$ ; man bearbeitet nacheinander jeden noch freien Knoten und muss im schlimmsten Fall alle  $|E|$  Kanten betrachten. Nach einer Idee von Hopcroft und Karp lässt sich die Laufzeit auf  $O(\sqrt{|V||E|})$  verbessern, indem man in maximal  $\sqrt{|V|}$  Iterationen nicht nur jeweils einen  $M$ -erweiternden Pfad, sondern mehrere auf einmal. Pseudocode und Implementierungen dieser Algorithmen lassen sich leicht finden.

Interessant (weniger aus algorithmischer als aus graphentheoretischer Sicht) ist auch die Charakterisierung eines  $C$ -perfekten Matchings (in dem also jeder  $C$ -Knoten gematcht ist) durch den Satz von Hall, der auch als Heiratssatz bekannt ist: Ein  $C$ -perfektes Matching existiert genau dann, wenn für alle Teilmengen  $X \subseteq C$  gilt, dass  $|N(X)| \geq |X|$  ist. Dabei ist  $N(X)$  die Nachbarschaft von  $X$ , also die Menge aller Knoten, die zu einem Knoten in  $X$  adjazent sind. Es ist klar, dass dies eine notwendige Bedingung ist: Ist für irgend eine Menge  $X$  einmal  $|N(X)| < |X|$ , kann  $X$  als Menge nicht "verheiratet" (gematcht) werden, da es nicht genügend potenzielle Partner gibt. Graphentheoretisch interessant ist, dass die Bedingung auch hinreichend ist.

**Beispiel zur Identifizierbarkeit.** Wir untersuchen die Identifizierbarkeit des Nullmusters

$$Z = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

mit 6 Sensoren und 4 Signalen. Wir erstellen für jede Spalte  $j$  den Graphen  $\tilde{G}_j$  zu  $\tilde{Z}_j$  und stellen fest, dass Spalten 1, 3 und 4 identifizierbar sind, Spalte 2 jedoch nicht; das größte Matching hat dort nur die Größe 2 statt der nötigen 3. Spalte 2 der Matrix  $M$  hat also einen freien Parameter.

### 4.3 Optimale Sensoren-Auswahl

Die Eigenschaft der Identifizierbarkeit ist von fundamentaler Bedeutung, wenn man aus einer Reihe von Experimenten sowohl die Mischungskoeffizienten  $A$  als auch die eigentlichen

Signale  $X$  aus (verrauschten) gemischten Messungen  $Y \approx AX$  schätzen will (bis auf die Skalierung). Ist bekannt, dass  $A$  ein identifizierbares Nullmuster hat, so ist dies grundsätzlich möglich. Man möchte daher nur mit solchen identifizierbaren Systemen arbeiten.

Man überlegt sich (Übung), dass das Hinzufügen (irgendwie verbundener) neuer Sensoren zu einem identifizierbaren System nicht schädlich ist; dies scheint intuitiv richtig. Formal beweist man: Ist ein Graph  $G = (C \cup R, E)$  gegeben,  $R' \subset R$  und der durch  $C \cup R'$  induzierte Teilgraph bereits identifizierbar, dann ist auch  $G$  identifizierbar.

Dies führt auf ein interessantes Optimierungsproblem: Gegeben sind Signalquellen (die wir nicht beeinflussen können) und eine (große) Menge an potenziellen Sensoren, die jeweils verschiedene Mengen Quellen mesen. Das Problem besteht darin, eine minimale (oder minimal teure) Menge von Sensoren so auszuwählen, dass das System identifizierbar ist.

**4.8 Problem (MINSENSOR).** Gegeben sei ein bipartiter Graph  $G = (C \cup R, E)$  (äquivalent: ein Nullmuster  $Z$ ) und eine Kostenfunktion  $cost : R \rightarrow \mathbb{R}$  (äquivalent: die jeder Zeile von  $Z$  reellwertige Kosten zuordnet). Gesucht ist eine Teilmenge  $Y \subseteq R$  (äquivalent: Menge von Zeilen in  $Z$ ), so dass der von  $C \cup Y$  induzierte Teilgraph (äquivalent: die Teilmatrix von  $Z$  mit den Zeilen aus  $Y$ ) identifizierbar ist und die Kosten  $\sum_{i \in Y} cost(i)$  minimal sind. •

**4.9 Satz.** *MINSENSOR ist NP-schwer.*

Solange man keine spezifischeren Ideen und Algorithmen hat, bietet es sich bei einem NP-schweren Problem an, eine Formulierung als ganzzahliges lineares Programm (integer linear program, ILP) oder gemischtes (mixed) ILP (MILP) zu versuchen. Das ILP kann man dann für hinreichend kleine Instanzen mit Standard-Software lösen lassen und durch problemspezifisches Wissen und Standard-Tricks weiter verbessern. Aus diesem Grund entwickeln wir eine solche Formulierung.

Am natürlichsten ist es, die Auswahl der Menge  $Y$  (Zeilen von  $Z$ ) mit Hilfe von Indikatorvariablen  $y = (y_i) \in \{0, 1\}^m$  zu beschreiben, so dass also  $y_i = 1 \Leftrightarrow i \in Y$  gilt. Die zu minimierende Zielfunktion lautet dann einfach  $\sum_{i=1}^m cost(i) \cdot y_i$ . Die Herausforderung ist, die Eigenschaft der Identifizierbarkeit mit Hilfe von linearen Gleichungen und Ungleichungen auszudrücken. Wir bauen dies in drei Schritten auf:

1. ein LP, um die Existenz eines  $C$ -perfekten Matchings in einem Graphen zu beschreiben,
2. ein LP, um die Identifizierbarkeit eines Nullmusters zu beschreiben,
3. ein LP, um die Identifizierbarkeit einer Zeilenauswahl eines Nullmusters zu beschreiben.

**Schritt 1: Existenz eines  $C$ -perfekten Matchings.** Es sei  $N(i) \subseteq C$  die Nachbarschaft von  $R_i$  und  $N(j) \subseteq R$  die Nachbarschaft von  $C_j$ . Wir führen für jede Kante  $\{R_i, C_j\}$  des Graphen eine Indikatorvariable  $m_{ij} \in \{0, 1\}$  ein, wobei  $m_{ij} = 1$  besagen soll, dass die Kante zu einem  $C$ -perfekten Matching gehört. Die offensichtlichen Bedingungen sind nun, dass jeder  $C$ -Knoten genau eine inzidente Kante haben muss und jeder  $R$ -Knoten höchstens eine inzidente Kante haben darf, also

$$\sum_{i \in N(j)} m_{ij} = 1 \quad \forall j \in C, \quad (4.1)$$

$$\sum_{j \in N(i)} m_{ij} \leq 1 \quad \forall i \in R. \quad (4.2)$$

Statt der (starken) Bedingung  $m_{ij} \in \{0, 1\}$  fordern wir lediglich

$$m_{ij} \geq 0 \quad \forall \text{ existierenden } (i, j). \quad (4.3)$$

Aus (4.2) und (4.3) folgt zusammen  $m_{ij} \in [0, 1]$  für alle existierenden  $(i, j)$ .

Interessanterweise gilt nun: Die Ecken des durch (4.1) bis 4.3 beschriebenen Polygons haben ganzzahlige Koordinaten. Das bedeutet: Ist die Menge der  $m_{ij}$ , die die Gleichungen und Ungleichungen erfüllen nicht leer, so liegt auch ein Punkt mit ganzzahligen Koordinaten darin (und ist eine Ecke). Daher kann man die Forderung  $m_{ij} \in \{0, 1\}$  hier durch die schwächere Forderung  $m_{ij} \in [0, 1]$  ersetzen und hat nur ein LP statt eines ILP zu lösen, was in polynomieller Zeit möglich ist. Der Grund liegt darin, dass die Koeffizientenmatrix  $A$  dieses linearen Erfüllbarkeitsproblems, wenn man die Menge der zulässigen  $(m_{ij})$  in Standardform  $\{m \mid m \geq 0, Am \leq b\}$  bringt, *total unimodular* ist. Das bedeutet, dass die Determinante jeder quadratischen Submatrix von  $A$  einen der Werte  $-1, 0 + 1$  annimmt.

**Schritt 2: Identifizierbarkeit eines Nullmusters.** Wir müssen sicherstellen, dass jede Spalte  $s$ , des Nullmusters identifizierbar ist, also alle Graphen  $\tilde{G}_s$  ein  $C$ -perfektes Matching haben. Dazu müssen wir  $n$  Varianten des LPs aus Schritt 1 erstellen, eine für jedes  $\tilde{G}_s$ , und diese zu einem großen LP kombinieren. Für jede in  $\tilde{G}_s$  existierende Kante führen wir daher Variablen  $m_{ij}^s$  ein. Insgesamt ergibt sich das: Finde  $m = (m_{ij}^s)$ , so dass

$$\sum_{i \in N(j) \setminus N(s)} m_{ij}^s = 1 \quad \forall s \in C, \forall j \in C \setminus \{s\}, \quad (4.4)$$

$$\sum_{j \in N(i) \setminus \{s\}} m_{ij}^s \leq 1 \quad \forall s \in C, \forall i \in R \setminus N(s), \quad (4.5)$$

$$m_{ij}^s \geq 0 \quad \forall s \in C, \forall j \in C \setminus \{s\}, \forall i \in N(j) \setminus N(s). \quad (4.6)$$

Die Korrektheit dieses LP (immer noch sind alle Variablen reellwertig) folgt sofort aus der von dem LP aus Schritt 1.

**Schritt 3: Auswahl von Zeilen.** Der springende Punkt ist nun, dass wir nicht fordern wollen, dass ganz  $Z$  identifizierbar ist, sondern die Teilmatrix, die durch die binären Indikatorvariablen  $y_i$  ausgewählt wurde, wobei wir deren Gesamtkosten minimieren wollen. Für die Zulässigkeit einer Lösung bedeutet das, dass wir  $z_{ij}^s$  nur dann auf 1 setzen dürfen, wenn  $y_i = 1$  ist. Es ergibt sich das folgende MILP in den Variablen  $(y, m)$ , dessen Korrektheit nun offensichtlich ist:

$$\text{Min. } \sum_{i \in R} \text{cost}(i) \cdot y_i, \quad \text{so dass} \quad (4.7)$$

$$\sum_{i \in N(j) \setminus N(s)} m_{ij}^s = 1 \quad \forall s \in C, \forall j \in C \setminus \{s\}, \quad (4.8)$$

$$\sum_{j \in N(i) \setminus \{s\}} m_{ij}^s \leq 1 \quad \forall s \in C, \forall i \in R \setminus N(s), \quad (4.9)$$

$$m_{ij}^s \leq y_i \quad \forall s \in C, \forall j \in C \setminus \{s\}, \forall i \in N(j) \setminus N(s), \quad (4.10)$$

$$m_{ij}^s \geq 0 \quad \forall s \in C, \forall j \in C \setminus \{s\}, \forall i \in N(j) \setminus N(s), \quad (4.11)$$

$$y_i \in \{0, 1\} \quad \forall i \in R. \quad (4.12)$$

Wir fassen die gemachten Aussagen in einem Satz zusammen (ohne formalen Beweis).

**4.10 Satz.** *Es ist  $(y, m)$  genau dann ganzzahlig und zulässig (d.h., erfüllt alle Gleichungen und Ungleichungen), wenn die  $m_{ij}^s$  in jedem Graphen  $\tilde{G}_s$  ein  $C$ -perfektes Matching beschreiben, das nur Kanten enthält, die inzident zu  $R$ -Knoten aus  $\{R_i \mid y_i = 1\}$  sind.*

*Existiert ein zulässiger Punkt  $(y, m)$  mit nicht ganzzahligem  $m$ , dann existiert mindestens ein zulässiger Punkt  $(y, m')$  mit ganzzahligem  $m'$  und mindestens genauso gutem Zielfunktionswert.*

**MILP-Engineering.** Die jetzige Formulierung kann durch freie oder kommerzielle MILP-Solver für nicht zu große Instanzen gelöst werden. Allein jedoch das ‐Aufstellen‐ der Koeffizientenmatrix der Ungleichungen benötigt Zeit und Platz. Daher haben sich sogenannte *cutting plane* Ansätze bewährt. Hier wird zunächst eine relaxierte Version des Problems gelöst, indem Bedingungen weggelassen oder durch schwächere ersetzt werden. Eine solche Lösung ist nicht notwendig zulässig für das Originalproblem, aber sie könnte es zufälligerweise sein. Ist sie zulässig, hat man auch das Originalproblem gelöst. Ansonsten muss man mindestens eine (Un)Gleichung des Originalproblems identifizieren, die verletzt ist, zum Problem hinzufügen und erneut lösen. Diese hinzugefügte (Un)Gleichung separiert die gefundene unzulässige Lösung von der Menge der zulässigen Lösungen; die durch sie gegebene Hyperebene ist eine ‐cutting plane‐.

Die Hoffnung bei diesem Vorgehen ist, dass man (a) einen effizienten Algorithmus für das Separierungsproblem hat, also eine verletzte Ungleichung effizient finden kann, insbesondere möglichst effizienter als alle Originalungleichungen durchzugehen und zu testen; und (b) nach wenigen Iterationsschritten bereits auf eine zulässige Lösung stößt. Es kann passieren, dass man so viele Iterationsschritte benötigt, dass man zum Schluß doch (fast) das ganze MILP betrachtet hat. In dem Fall hätte man nichts gewonnen, sondern erheblichen Mehraufwand betrieben, und es wäre besser gewesen, das ganze MILP direkt zu lösen. In der Praxis zeigt sich aber, dass (bei guten problemspezifischen Relaxierungen und Separierungsverfahren) viel zu gewinnen ist.

Ein weiterer Trick ist, auch zunächst nicht alle Variablen in das MILP aufzunehmen, sondern nur eine essentielle Teilmenge, und zusammen mit den verletzten Ungleichungen die benötigten Variablen hinzuzufügen. Man spricht dabei von *column generation*, da Variablen den Spalten in der Koeffizientenmatrix des ((M)I)LPs entsprechen.

Hier verzichten wir *ganz* auf die  $m_{ij}^s$ -Variablen und benutzen zu Beginn statt (4.8)–(4.11) zunächst einfache notwendige Bedingungen für die  $y_i$ : Das System mit durch  $(y_i)$  gewählten Sensoren ist höchstens dann identifizierbar, wenn

1. für alle  $s \in C$  gilt, dass  $s$  mindestens ausgewählt  $|C| - 1 = n - 1$  Nichtnachbarn hat, denn nur Nichtnachbarn bleiben ja in  $\tilde{G}_s$  überhaupt übrig; es muss also gelten:

$$\sum_{i \in R \setminus N(s)} y_i \geq n - 1 \quad \forall s \in C;$$

2. für alle  $s \neq s'$  gilt, dass nicht  $N(s) \cap Y \subseteq N(s') \cap Y$  ist. Wäre dies nämlich der Fall, könnte Knoten  $s$  im Graphen  $\tilde{G}_{s'}$  durch die  $Y$ -Knoten nicht mehr gematcht werden,

denn alle Nachbarn von  $s'$  sind ja nicht mehr vorhanden, also hätte auch  $s$  keine ausgewählten Nachbarn mehr. Also muss gelten:

$$\sum_{i \in N(s) \setminus N(s')} y_i \geq 1 \quad \forall s_1 \neq s_2.$$

Obwohl auch diese Variante schon NP-schwer ist, ist sie doch viel einfacher als das ursprüngliche Problem; es gibt nur die  $y$ -Variablen und  $|C|^2$  Ungleichungen.

Sobald wir nun eine Lösung  $y \in \{0, 1\}^{|R|}$  des relaxierten Problems erhalten haben, berechnen wir für jedes  $s \in C$  ein größtes Matching mit den ausgewählten  $R$ -Knoten in  $\tilde{G}_s$ . Haben alle die Größe  $|C| - 1$ , ist die Lösung  $y$  zulässig und wir sind fertig. Wenn das größte Matching für ein  $s$  nicht die nötige Größe hat, dann gibt es nach dem Satz von Hall (Heiratsatz) dort eine Menge  $X \subseteq C \setminus \{s\}$  mit  $|X| > |N(X) \cap (Y \setminus N(s))|$ . In der Tat liefert uns das größte Matching gewissermaßen gratis eine solche Menge  $X$ , die ein  $C$ -perfektes Matching verhindert, als Zugabe (Übung). Wir fügen nun eine entsprechende Ungleichung zum ILP hinzu, die dafür sorgt, dass die Menge  $X$  "versorgt" wird:

$$\sum_{i \in N(X) \setminus N(s)} y_i \geq |X|.$$

Damit haben wir ein weniger relaxiertes Problem erhalten, das nun erneut gelöst werden kann; hierbei werden die  $m_{ij}^s$ -Variablen nie eingeführt. Da es  $2^{|C|-1}$  mögliche Teilmengen  $X$  gibt, die Hall's Theorem verletzen könnten, ist es denkbar, dass ebenso viele (immer weniger relaxierte) ILPs gelöst werden müssen. In der Praxis zeigt sich jedoch, dass dieses Vorgehen gut funktioniert und nach wenigen Iterationen eine zulässige Lösung gefunden wird.

## 4.4 Separable Signale

Wir kommen nun zu einer ganz anderen Frage. Als Motivation stellen wir uns vor, dass wir (aus Zeit- oder Kostengründen) nicht beliebig viele unabhängige Experimente durchführen können;  $p$  ist also beschränkt. Das führt möglicherweise dazu, dass wir nicht mehr alle Signale messen können oder wollen. Eine interessante Frage ist also, gibt es eine Teilmenge  $J \subseteq C$ , die unabhängig von den anderen Signalen gemessen werden kann und idealerweise noch identifizierbar ist? Damit wir  $J$  isoliert betrachten können, müssen wir eine Menge  $I \subseteq R$  an Sensoren finden, die zusammen genau die Signale aus  $J$  messen; es muss also  $N(I) = J$  sein;  $N()$  bezeichnet wieder die Menge der Nachbarn. Achtung:  $N(J) = I$  ist *nicht* gefordert!

**4.11 Definition** (Separable Signalmenge). Eine Signalmenge  $J \subseteq C$  heißt *separabel*, wenn es eine Sensormenge  $I \subseteq R$  gibt mit  $N(I) = J$ .

Gibt es sogar  $I_1, I_2$  mit  $N(I_1) = J$  und  $N(I_2) = J$ , dann ist auch  $N(I_1 \cup I_2) = J$ . Deswegen gibt es, sofern  $J$  überhaupt separabel ist, eine eindeutige größte Menge  $I$  mit  $N(I) = J$ . Diese nennen wir  $S(J)$ ; dies ist nur definiert, wenn  $J$  separabel ist.

Wie kann man testen, ob eine gegebene Menge  $J$  separabel ist und gegebenenfalls  $S(J)$  finden? Dies ist einfach: Man beginnt mit der leeren Menge  $I$ , iteriert über alle  $R$ -Knoten  $R_i$  und fügt  $R_i$  zu  $I$  hinzu, wenn  $N(R_i) \subseteq J$  ist. Am Ende der Iteration ist entweder  $N(I) = J$

oder  $N(I) \subsetneq J$ . Im ersten Fall ist  $J$  separabel und  $I = S(J)$ ; im zweiten Fall ist  $J$  nicht separabel.

Ist  $J$  separabel, kann man fragen, ob es eine Sensorauswahl gibt, die das Netzwerk, eingeschränkt auf die Signale in  $J$ , identifizierbar macht. Da  $S(J)$  die maximal mögliche Sensormenge darstellt, ist dies genau dann der Fall, wenn der von  $J$  und  $S(J)$  induzierte Teilgraph identifizierbar ist. (Man kann dann noch wie in Abschnitt 4.3 beschrieben eine minimale Sensormenge auswählen.)

**4.12 Definition** (Sauber separable Signalmenge). Eine Signalmenge  $J \subseteq C$  heißt *sauber separabel*, wenn sie separabel ist und der durch  $J$  und  $S(J)$  induzierte Teilgraph von  $G = (R \cup C, E)$  identifizierbar ist.

Offensichtlich ist die leere Menge stets identifizierbar, aber uninteressant. Interessante Fragen sind beispielsweise die nach der kleinsten nichtleeren sauber separablen Menge, oder die nach der Existenz einer sauber separablen Menge der Größe in einem gegebenen Intervall.

Wiederum lassen sich solche Probleme als MILP formulieren. Wir bauen auf dem MINSENSOR-MILP auf, benötigen nun aber zusätzliche Indikatorvariablen  $x = (x_j)_{j \in C} \in \{0, 1\}^{|C|}$ , die anzeigen, welche Signale ausgewählt wurden. Die Variablen  $y = (y_i) \in \{0, 1\}^m$  stellen die Menge  $I$  dar. Die Bedingung  $N(I) = J$  wird wie folgt formuliert: Jedes  $i \in R$ , für das ein  $j \in N(i) \setminus J$  existiert, darf nicht ausgewählt werden.

---

# Analyse von Reaktionsnetzwerken

---

## 5.1 Einleitung

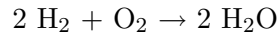
Metabolische Prozesse (Umwandlung von beispielsweise mit der Nahrung aufgenommenen Stoffen in körpereigene Substanzen, Speicherung und Freigabe von Energie) in lebenden Zellen sind nichts anderes als (stark miteinander verwobene) chemische Reaktionen, lassen sich also durch Reaktionsnetzwerke beschreiben. Wir stellen in diesem Kapitel Methoden bereit, um Reaktionsnetzwerke zu beschreiben und zu analysieren.

Grundlage ist dabei die stöchiometrische Matrix  $S$ , die wir in Abschnitt 5.2 einführen.

## 5.2 Die stöchiometrische Matrix

Die stöchiometrische Matrix  $S$  beschreibt, welche chemischen Verbindungen (compounds) in welchen Reaktionen miteinander reagieren und welche neuen Verbindungen dabei gebildet werden. Es ist Konvention, dass die  $m$  Zeilen die Verbindungen (Moleküle, Metabolite) repräsentieren, und jede der  $n$  Spalten eine Reaktion darstellt. In komplexen Reaktionsnetzwerken ist in der Regel die Anzahl der Reaktionen größer als die Anzahl der beteiligten Moleküle ( $n > m$ ). Der Eintrag  $S_{ij}$  beschreibt, wie viele Moleküle des Typs  $i$  in der  $j$ -ten Reaktion entstehen. Wird Verbindung  $i$  in der Reaktion verbraucht, so ist  $S_{ij}$  negativ.

**5.1 Beispiel** (Bildung von Wasser). Ein Beispiel für eine einfache (nicht metabolische) chemische Reaktion ist die Verbindung von Wasserstoff und Sauerstoff zu Wasser. Dabei reagieren zwei Wasserstoffmoleküle, die aus jeweils zwei Wasserstoffatomen bestehen mit einem Sauerstoffmolekül, das aus zwei Sauerstoffatomen besteht, zu zwei Wassermolekülen, die jeweils aus zwei Wasserstoff- und einem Sauerstoffatom bestehen:



Es gibt drei Verbindungen und nur eine Reaktion. Die stöchiometrische Matrix hat die Größe  $3 \times 1$  und lautet  $S = \begin{pmatrix} -2 \\ -1 \\ 2 \end{pmatrix}$ , wobei die Verbindungen in der Reihenfolge  $\text{H}_2$ ,  $\text{O}_2$ ,  $\text{H}_2\text{O}$  angegeben sind. Die Rückreaktion (Aufspaltung von Wasser in seine Elemente) ist nur unter großem Energieaufwand möglich. Betrachtet man diese Reaktion im gegebenen Netzwerk, so würde eine zweite Spalte, die das Negative der ersten Spalte ist, hinzukommen. Hieran sieht man schon, dass stöchiometrische Matrizen in der Regel keinen vollen Rang haben. ♡

Typischerweise gilt in großen Reaktionsnetzwerken also:

$$\text{rang}(S) < m = \text{Anzahl der Verbindungen} < n = \text{Anzahl der Reaktionen.}$$

Tatsächlich ist Beispiel 5.1 nicht sehr gut, da diese Reaktion zu “elementar” ist. In den Reaktionsnetzwerken, die wir betrachten, finden Reaktionen auf komplexeren Molekülen statt, sogenannten *Metaboliten*, die mit Hilfe von *Enzymen* zu anderen Metaboliten umgebaut werden. Eine besondere Rolle als Signalmoleküle spielen dabei Phosphatgruppen, die einem Metabolit (oft auch mehrfach) hinzugefügt werden können. Die meisten Reaktionen lassen sich in drei Gruppen klassifizieren. Wir gehen hier davon aus, dass alle Reaktionen reversibel sind (also durch zwei Spalten mit jeweils inversen Vorzeichen in  $S$  repräsentiert werden); dies ist der Regelfall. Dabei bezeichne im Folgenden  $C$  einen Metaboliten,  $P$  eine Phosphatgruppe und  $A$  einen Kofaktor (Überträger von  $P$ ); dementsprechend bezeichnen  $CP$ ,  $PC$ ,  $AP$  verschiedene Verbindungen aus  $C$  und  $P$  bzw.  $A$  und  $P$ .

1. Bimolekulare Assoziation:  $C + P \rightleftharpoons CP$ ; zwei Moleküle werden “verbraucht” und ein neues Molekül entsteht.
2. Kofaktor-gekoppelte Reaktion:  $C + AP \rightleftharpoons CP + A$ ; Phosphat wird vom Träger  $A$  auf  $C$  übertragen. Diese Reaktion lässt sich auch mit Hilfe von zwei bimolekularen Assoziationen schreiben.
3. Reversible Konversion:  $CP \rightleftharpoons PC$ ; ein Molekül wird reversibel zu einem anderen umgebaut.

Die Anzahl der beteiligten Moleküle muss nicht immer eins sein. Für die drei Reaktionsklassen sieht ein Ausschnitt von  $S$  folgendermaßen aus (die Spalten stellen die Reaktionen in der angegebenen Reihenfolge dar).

$$S = \left( \begin{array}{c|cccccc} C & -1 & 1 & -1 & 1 & 0 & 0 \\ P & -1 & 1 & 0 & 0 & 0 & 0 \\ CP & 1 & -1 & 1 & -1 & -1 & 1 \\ A & 0 & 0 & 1 & -1 & 0 & 0 \\ AP & 0 & 0 & -1 & 1 & 0 & 0 \\ PC & 0 & 0 & 0 & 0 & 1 & -1 \end{array} \right)$$

Man kann die Matrix  $S$  zeilen- oder spaltenweise anschauen, also:

- Für eine gegebene Verbindung (Zeile): An welchen Reaktionen (Spalten) ist sie (wie) beteiligt?

- Für eine gegebene Reaktion (Spalte): Welche Verbindungen (Zeilen) sind an ihr (wie) beteiligt?

Ersetzen wir in  $S$  jeden von Null verschiedenen Eintrag durch 1, erhalten wir eine Inzidenzmatrix, die beschreibt, welche Verbindungen an welchen Reaktionen beteiligt sind (aber nicht, wie oder in welcher Menge). Die so erhaltene Matrix nennen wir *Inzidenzmatrix* des Reaktionsnetzwerks und bezeichnen sie mit  $\hat{S}$ . Ihre Analyse liefert Hinweise zur Netzwerkstruktur.

Der Beweis der folgenden Aussage ist eine Übungsaufgabe.

**5.2 Lemma.** Sei  $A^R := \hat{S}\hat{S}^T$  (eine symmetrische  $m \times m$ -Matrix). Dann ist  $A_{ii}^R$  die Zahl der Reaktionen, an denen Verbindung  $i$  beteiligt ist, und  $A_{ik}^R$  für  $i \neq k$  die Zahl der Reaktionen, an denen Verbindungen  $i$  und  $k$  gemeinsam beteiligt sind.

Sei  $A^C := \hat{S}^T\hat{S}$  (eine symmetrische  $n \times n$ -Matrix). Dann ist  $A_{jj}^C$  die Zahl der Verbindungen, die an Reaktion  $j$  beteiligt sind, und  $A_{jl}^C$  für  $j \neq l$  die Zahl der Verbindungen, die gemeinsam an Reaktionen  $j$  und  $l$  beteiligt sind.

Man nennt daher  $A^R$  *Adjazenzmatrix der Reaktionen* und  $A^C$  *Adjazenzmatrix der Verbindungen* (compounds).

Es kann von Interesse sein, in komplexen Reaktionsnetzwerken die Verteilung der Größenordnungen der Einträge von  $A^R$  und  $A^C$  zu betrachten (getrennt nach Diagonalelementen und Nichtdiagonalelementen).

### 5.3 Elemente und Erhaltungsgrößen

Die Verbindungen, die die Zeilen von  $S$  bilden, bestehen aus kleineren Bausteinen, z.B. den chemischen Elementen wie C,H,N,O,P,S und anderen. Ein Grundgesetz chemischer Reaktionen ist, dass dabei keine Elemente vernichtet werden oder aus dem Nichts entstehen. Das heißt, die Anzahl der Elemente jeder Art auf beiden Seiten einer Reaktionsgleichung muss identisch sein.

**5.3 Definition.** Sei  $p$  die Anzahl der auftretenden Elemente, sei  $m$  nach wie vor die Anzahl der auftretenden Verbindungen. Die Element-Matrix ist eine  $p \times m$ -Matrix  $E$ , in der  $E_{q,i}$  beschreibt, welche Menge des  $q$ -ten Elements in Verbindung  $i$  enthalten ist. Spalte  $i$  beschreibt also die Summenformel von Verbindung  $i$ .

**5.4 Beispiel** (Wasser und Ethanol). Wir betrachten die Moleküle  $H_2$ ,  $O_2$ ,  $H_2O$  und Ethanol  $C_2H_5OH$  aus den Elementen H, O und C (jeweils in dieser Reihenfolge). Dies liefert die  $3 \times 4$ -Matrix

$$E = \left( \begin{array}{c|cccc} \text{H} & 2 & 0 & 2 & 6 \\ \text{O} & 0 & 1 & 1 & 1 \\ \text{C} & 0 & 0 & 0 & 2 \end{array} \right).$$

♡

**5.5 Lemma.** Sei  $E$  eine  $p \times m$  Element-Matrix, und sei  $S$  eine  $m \times n$  stöchiometrische Matrix mit denselben Verbindungen wie  $E$ . Dann ist  $ES = 0_{p \times n}$ .

**Beweis.** Die Aussage  $ES = 0$  ist genau die Balanciertheitsbedingung der Elemente in allen chemischen Reaktionen: Betrachten wir die Aussage für Zeile (Element)  $q$  von  $E$ ; die Behauptung ist  $e_q \cdot S = 0_{1 \times n}$ . Das heißt aber nichts anderes als  $\sum_{i=1}^m E_{q,i} \cdot S_{i,j} = 0$  für alle  $j$ , also dass Element  $q$  in Reaktion  $j$  insgesamt eine Bilanz von Null aufweist. Mit anderen Worten:  $S$  wäre keine zulässige stöchiometrische Matrix, wenn diese Bedingung nicht gelten würde.  $\square$

Statt chemischer Elemente im strengen Sinn können auch andere in chemischen Reaktionen erhaltene Quantitäten in die Element-Matrix aufgenommen werden, z.B. Ladung. Die erlaubt, auch beispielsweise protoniertes Wasser  $\text{H}_3\text{O}^+$  als Molekül zu beschreiben; die Ladung  $+1$  wird wie ein zusätzliches Element behandelt.

Aus Lemma 5.5 folgt: Jedes chemische Element (entsprechend einer Zeile  $e_q$  von  $E$ ) ist eine Erhaltungsgröße des Reaktionsnetzwerks wegen

$$\begin{aligned} e_q \cdot S &= 0_{1 \times n} && \text{oder} \\ S^T \cdot e_q^T &= 0_{n \times 1} && \text{oder} \\ e_q^T &\in \text{Kern}(S^T). \end{aligned}$$

Es kann allerdings noch mehr Vektoren im Kern von  $S^T$  geben als die chemischen Elemente. Diese Erkenntnis nehmen wir zum Anlass,  $\text{Kern}(S^T)$  den *Raum der Erhaltungsgrößen* der Moleküle von  $S$  zu definieren. Erinnerung:  $\text{Kern}(A) = \{x \mid Ax = 0\}$ .

Dies ist lediglich einer von vier wichtigen Unterräumen. Zu den anderen kommen wir im Folgenden. Anschließend geben wir eine Übersicht über alle vier wichtigen Unterräume.

## 5.4 Fluxe und Reaktionsdynamik

Die Hauptfunktion von  $S$  ist nicht die Beschreibung der Netzwerktopologie oder von Erhaltungsgrößen, sondern die Beschreibung der möglichen Dynamik von Reaktionen. Dazu müssen wir den Begriff des Fluxes einführen.

Wir beschreiben zunächst eine (unrealistische) zeitdiskrete Version, die für die intuitive Vorstellung hilfreich ist. Anschließend beschreiben wir das (im weiteren Verlauf ausschließlich vorkommende) kontinuierliche Modell.

Angenommen, jede der Reaktionen läuft synchron im Takt mit den anderen Reaktionen ab. Sei  $x(t)$  der  $m$ -Vektor, der die vorhandenen Mengen jeder Verbindung zur Zeit  $t$  beschreibt. Der diskrete Flux-Vektor  $f$  ist ein Vektor der Länge  $n$  und beschreibt, wie oft jede Reaktion im betrachteten Takt stattfindet. Es ist klar, dass die Gesamtänderung der Menge jeder Verbindung dann durch  $Sf$  gegeben ist. Wir können sogar zulassen, dass  $f = f(t, x)$  von  $t$  und von  $x$  abhängt. Dann ist  $x(t+1) - x(t) = S \cdot f(t, x(t))$ , und das System lässt sich bei gegebenen Anfangswerten Schritt für Schritt simulieren.

In Wahrheit sind Reaktionen weder gleich getaktet noch gleich schnell. In Wahrheit beschreibt der (kontinuierliche) Flux  $v$  daher die Reaktionsintensitäten pro Zeiteinheit für die  $n$  Reaktionen, so dass aus obiger Differenzgleichung die Differenzialgleichung

$$\dot{x}(t) = S \cdot v$$

wird: Die Änderungsraten der Verbindungskonzentrationen ergeben sich aus dem Produkt der stöchiometrischen Matrix mit den Reaktionsraten (Fluxen). Wählt man  $v = e_j$ , den  $j$ -ten Einheitsvektor, wird nur die  $j$ -te Reaktion (mit einer Einheitsrate) ausgeführt, und  $Sv = Se_j$ , die  $j$ -te Spalte von  $S$ , gibt die Änderungsraten der Molekülkonzentrationen für diese Reaktion an.

Von besonderem Interesse sind Fluxe (Kombinationen von Reaktionen), für die das Reaktionsnetzwerk stationär ist, wo also  $\dot{x}(t) = 0$ , also  $Sv = 0$ , also  $v \in \text{Kern}(S)$  gilt. Damit haben wir  $\text{Kern}(S)$  als Unterraum aller stationären Fluxe des Systems erkannt.

## 5.5 Vier durch die stöchiometrische Matrix definierte Unterräume

Gegeben sei die stöchiometrische Matrix  $S$ . Wir wissen bereits:

$$\begin{aligned} \text{Kern}(S) \subset \mathbb{R}^m &\text{ ist der Unterraum der stationären Fluxe,} \\ \text{Kern}(S^T) \subset \mathbb{R}^n &\text{ ist der Unterraum der Erhaltungsgrößen.} \end{aligned}$$

Wir erinnern an die Definition des orthogonalen Komplements eines Unterraums  $U$ : Das *orthogonale Komplement*  $U^\perp$  von  $U$  ist die Menge aller Vektoren, die auf allen Vektoren aus  $U$  senkrecht stehen, also

$$U^\perp := \{ v \mid u^T v = 0 \text{ für alle } u \in U \}.$$

Aus der linearen Algebra wissen wir weiter:

$$\begin{aligned} \text{Bild}(S^T) &= \text{Kern}(S)^\perp, \\ \text{Bild}(S) &= \text{Kern}(S^T)^\perp. \end{aligned}$$

Aus der ersten Identität folgt, dass sich jeder Fluxvektor  $v$  eindeutig orthogonal zerlegen lässt in einen stationären Teil  $v_{\text{stat}}$  und einen dynamischen Teil  $v_{\text{dyn}}$ , so dass

$$v = v_{\text{stat}} + v_{\text{dyn}}, \quad v_{\text{stat}} \in \text{Kern}(S), \quad v_{\text{dyn}} \in \text{Bild}(S^T),$$

der dynamische Teil ist also eine Linearkombination der Zeilen von  $S$ . Eine Zeile von  $S$  zeigt, wie sich ein Molekül in allen Reaktionen verhält.

$\text{Bild}(S)$  ist die Menge aller Linearkombinationen der Spalten (Reaktionen) von  $S$  und stellt damit alle möglichen Konzentrationsänderungsraten von Molekülen des Systems dar. Wie bereits gesagt ist das orthogonale Komplement dazu der Raum der Erhaltungsgrößen.

Es ist für die Analyse eines Systems  $S$  von Interesse, Bild und Kern von  $S$  und von  $S^T$  zu berechnen. Das bedeutet, eine Basis für diese Unterräume anzugeben. Dies ist prinzipiell kein Problem; das Gauss'sche Eliminationsverfahren zur Lösung eines Gleichungssystems  $Ax = b$  liefert "nebenbei" eine Basis für  $\text{Kern}(A)$ .

Am elegantesten ist es, eine Singulärwertzerlegung von  $S$  zu berechnen. Sei

$$S = U\Sigma V^T$$

eine Singulärwertzerlegung von  $S \in \mathbb{Z}^{m \times n}$  mit  $U = (u_1, \dots, u_m) \in \mathbb{R}^{m \times m}$ ,  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{m \times n}$  vom Rang  $r$  und  $V = (v_1, \dots, v_n) \in \mathbb{R}^{n \times n}$ . Dann ist

$$\begin{aligned} Sv_j &= \sigma_j u_j, \\ S^T u_i &= \sigma_i v_i. \end{aligned}$$

Damit ist klar:

1. Eine orthonormale Basis für  $\text{Kern}(S)$  ist  $\{v_{r+1}, \dots, v_n\}$ .
2. Eine orthonormale Basis für  $\text{Bild}(S)$  ist  $\{u_1, \dots, u_r\}$ .
3. Eine orthonormale Basis für  $\text{Kern}(S^T)$  ist  $\{u_{r+1}, \dots, u_m\}$ .
4. Eine orthonormale Basis für  $\text{Bild}(S^T)$  ist  $\{v_1, \dots, v_r\}$ .

## 5.6 Interne und Externe Reaktionen

Bei der bisherigen Charakterisierung der durch  $S$  definierten Unterräume haben wir angenommen, dass die Matrix  $S$  vollständig ein Reaktionsnetzwerk beschreibt.

Tatsächlich betrachtet man jedoch fast immer nur einen Ausschnitt, d.h., es gibt Stoffe, die nicht mehr in der Matrix erfasst werden; Reaktionen, die diese Stoffe verbrauchen oder erzeugen, heißen *externe* Reaktionen; die anderen Reaktionen heißen *interne Reaktionen*. Entsprechend spricht man bei den Reaktionsraten von *internen Flüssen* und *externen Flüssen*.

Es ist sinnvoll, intern und extern in der Matrix zu trennen. Es gebe  $n_I$  interne und  $n_E$  externe Reaktionen; deren Flüsse bezeichnen wir mit  $v = (v_1, \dots, v_{n_I})$  und  $b = (b_1, \dots, b_{n_E})$ ; insgesamt haben wir den Flussvektor  $(v, b)^T$  der Länge  $n = n_I + n_E$ .

Die (mathematische) Behandlung der internen Flüsse  $v$  unter der externen Flüsse  $b$  unterscheidet sich geringfügig. Bei den internen Flüssen  $v$  setzen wir  $v \geq 0$  voraus (eine Reaktion kann nicht "rückwärts" ablaufen). Will man explizit eine reversible Reaktion modellieren, muss man zwei Reaktionen mit inversen Vorzeichen verwenden. Die externen Flüsse werden (sofern sie nicht reversiblen Reaktionen entsprechen) so gerichtet, dass ebenfalls  $b_j \geq 0$  für nicht reversible  $j$  gilt. Bei reversiblen externen Reaktionen wird im Unterschied zu reversiblen internen Reaktionen jedoch *keine* zweite Reaktion mit inversen Vorzeichen angelegt. Wir ordnen die Reaktionen oBdA so, dass  $b = (b_{\text{uni}}, b_{\text{rev}})^T$ .

Insgesamt haben wir also die Fluxvektoren  $(v, b_{\text{uni}}, b_{\text{rev}})^T$  mit  $v \geq 0$ ,  $b_{\text{uni}} \geq 0$ .

**5.6 Beispiel** (Kleines Netzwerk). Es gebe die 7 Stoffe  $A, B, C, D, E, \text{byp}, \text{cof}$  (dabei steht *byp* für ein Nebenprodukt, *cof* für einen Kofaktor), 6 interne und 3 externe Reaktionen.

$$S = \left( \begin{array}{c|cccccc|ccc} & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & b_1 & b_2 & b_3 \\ \hline A & -1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ B & 1 & -2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ C & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\ D & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 \\ \text{byp} & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ \text{cof} & 0 & 0 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \end{array} \right)$$

Alle externen Reaktionen seien nicht reversibel:  $b_{\text{uni}} = (b_1, b_2, b_3)$ ,  $b_{\text{rev}} = ()$ . Man versuche, hiervon eine graphische Darstellung zu zeichnen. ♡

Wir können noch (unabhängig von der “technischen” Unterscheidung in reversible und nicht-reversible externe Reaktionen) eine weitere Unterscheidung der externen Reaktionen vornehmen: nämlich solche, die für uns interessante primäre Metabolite betreffen und solche, die nur Währungsmetabolite betreffen. Im Beispiel seien  $A, B, C, D, E$  die interessanten primären Metabolite; davon sind nur  $A$  und  $E$  an externen Reaktionen beteiligt. Damit sind die durch die Fluxe  $b_1$  und  $b_2$  beschriebenen Reaktionen interessant; die durch  $b_3$  beschriebene Reaktion, bei der das Nebenprodukt *byp* aus dem System verschwindet, wäre “uninteressant”. Diese Klassifikation ist dem Anwender überlassen (wobei sich die Anwender meist einig sind, was “interessant” ist und was nicht).

## 5.7 Extreme Pathways

Mit Hilfe der Singulärwertzerlegung können alle vier relevanten Unterräume von  $S$  charakterisiert werden. Jedoch sind die so erhaltenen orthogonalen Basen nicht immer einfach interpretierbar.

**5.7 Beispiel** (Einfaches Reaktionsnetzwerk). Gegeben sind 4 Moleküle  $A, B, C, D$  und 6 Einwegreaktionen (nicht reversible Reaktionen):

$$S = \left( \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline A & 1 & -1 & 0 & 0 & -1 & 0 \\ B & 0 & 1 & -1 & 0 & 0 & 0 \\ C & 0 & 0 & 1 & -1 & 0 & 1 \\ D & 0 & 0 & 0 & 0 & 1 & -1 \end{array} \right)$$

Zunächst ist es sinnvoll, sich das Reaktionsnetzwerk und die Basen graphisch zu veranschaulichen. Zwei der Reaktionen sind extern (1 und 4). (Um den Regeln des vorigen Abschnitts zu entsprechen, müsste man die Reaktionen jetzt umordnen; für die Aussage dieses Beispiels ist die Ordnung jedoch unerheblich.) Eine SVD zeigt  $\dim \text{Bild}(S) = \text{rang}(S) = 4$  und  $\dim \text{Kern}(S) = 2$ . Hier sind zwei verschiedene Basen für  $\text{Kern}(S)$ ; beide sind nicht orthonormal, stammen also nicht aus der SVD:

$$B_1 = \begin{pmatrix} 1 & 0 \\ 1 & -1 \\ 1 & -1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}.$$

Wir sehen, dass es mit  $B_1$  ein Problem gibt: Der zweite Basisvektor enthält negative Fluxe; diese sind aber nicht zugelassen, da die Reaktionen nur in die gegebene Richtung ablaufen können. Bei  $B_2$  taucht das Problem nicht auf; diese Basis ist interpretierbar. ♡

Was wir benötigen, ist nicht eine Basis im üblichen Sinn (Vektorraumbasis), sondern eine Menge an Vektoren, so dass alle nichtnegativen stationären Fluxe als (nichtnegative)

Linearkombination dieser Vektoren geschrieben werden kann. Wir suchen also zur Menge  $X = \{p \mid Sp = 0, p \geq 0\}$  eine Menge von Vektoren  $p_k$ , so dass jedes  $p \in X$  darstellbar ist als

$$p = \sum_k \alpha_k p_k \quad \text{mit } \alpha_k \geq 0 \text{ für alle } k.$$

Die  $p_k$  heißen in diesem Zusammenhang *extreme Pathways* (*extreme Reaktionspfade*); sie begrenzen den Raum aller Fluxe, die überhaupt im System zulässig sind. Jeder zulässige Flux kann als (nichtnegative) Kombination der extreme Pathways geschrieben werden. Jeder extreme-Pathway-Flux  $p_k$  lässt sich wieder aufteilen in den internen Teil  $v_k$  und den externen Teil  $b_k$ ;  $p_k = (v_k, b_k)^T$ , wenn man die Reaktionen entsprechend geordnet hat.

Wir können die extreme Pathways in drei Typen einteilen:

**Typ I: primäre Systempfade** sind dadurch charakterisiert, dass bei den “interessanten” externen Flüssen Aktivität zu beobachten ist (es gibt in  $p_k$  “interessante”  $b_{kj} \neq 0$ ).

**Typ II: zwecklose Pfade** sind dadurch charakterisiert, dass zwar externe Flüsse gibt, aber keine “interessanten” (also gibt es in  $p_k$  zwar gewisse Indizes  $j$  mit  $b_{kj} \neq 0$ , für alle interessanten  $j$  gilt  $b_{kj} = 0$ ).

**Typ III: interne Pfade** sind dadurch charakterisiert, dass es gar keine externen Flüsse gibt ( $b_k = 0$ ). Hier wird ein interner Stoffkreislauf betrachtet. Ein besonders uninteressanter Spezialfall (den man sogar als Typ IV bezeichnen könnte) ist eine reversible interne Reaktion, die ja als zwei Reaktionen dargestellt wird. Nutzt man Vorwärts- und Rückwärtsreaktion gleichermaßen, passiert in der Summe nichts. Diese Zyklen existieren zwar und sind extreme Pathways, aber sind als Analyseergebnisse so trivial, dass wir sie normalerweise aus der Ausgabe einer solchen Analyse herausfiltern werden.

Die interessante Frage ist nun, wie wir von  $S$  (oder von einer schon berechneten einer Vektorraumbasis von  $\text{Kern}(S)$ ) zu den extreme Pathways kommen. Hierzu betrachten wir folgenden Algorithmus, der hier mit Erörterungen angegeben wird.

1. Wir bilden das  $(n+m) \times n$ -Tableau (die Matrix)  $T^0 := \begin{bmatrix} \text{Id}_n \\ S \end{bmatrix}$ . In mehreren Iterationsschritten (bei denen weitere Spalten erzeugt werden) wird nun jede Zeile des  $S$ -Teils bearbeitet und durch elementare Spaltenoperationen einer Nullmatrix transformiert. Dabei entstehen im oberen Id-Teil der Matrix die extreme Pathways als Spalten. Die Idee ist, dass stets folgende Invariante erhalten bleibt:

Eine Spalte des Tableaus (Länge  $n+m$ ) beschreibt eine (nichtnegative) Kombination von Reaktionen; sie zeigt in ihren oberen  $n$  Einträgen die (nichtnegativen) Koeffizienten der ursprünglichen Reaktionen für diese Kombination und in ihren unteren  $m$  Einträgen die Auswirkungen diese Kombination auf alle  $m$  Molekülkonzentrationen.

Die Initialisierung mit  $\text{Id}$  und  $S$  erfüllt offenbar die Invariante. Durch das Bilden von (nichtnegativen) Linearkombinationen von Spalten bleibt die Invariante erhalten.

2. Allerdings sind reversible und nichtreversible externe Reaktionen getrennt zu behandeln. Deswegen teilen zunächst wir die Spalten auf in das Tableau  $T = [T^0 \mid T^\bullet]$ , so dass  $T^\bullet$  den reversiblen Reaktionen entspricht. Nun hat  $T^0$  die Dimension  $(n+m) \times (n_I + n_{\text{uni}})$  und  $T^\bullet$  die Dimension  $(n+m) \times n_{\text{rev}}$ .

3. Wir betrachten nun solche Zeilen aus dem unteren Teil von  $T^0$ , für die  $T^\bullet$  bereits eine Nullzeile ist; das entspricht den Metaboliten, die an keiner reversiblen externen Reaktion teilnehmen. Es gebe  $M \leq m$  solcher Zeilen. (Die anderen Zeilen werden bei diesem Schritt übersprungen.) In Iteration  $1 \leq \mu \leq M$  wird die  $\mu$ -te solche Zeile  $i_\mu$  betrachtet. Dabei wird aus  $T^{\mu-1}$  ein neues Tableau  $T^\mu$  erstellt. In Iteration  $\mu$  (für Zeile  $i_\mu$ ):
- Erzeuge ein (zunächst leeres) neues Tableau  $T^\mu$ .
  - Sei  $t$  die  $i_\mu$ -te Zeile des alten Tableaus  $T^{\mu-1}$ . Definiere eine Zerlegung der Spaltenindizes:  $J_- := \{j \mid t_j < 0\}$ ,  $J_0 := \{j \mid t_j = 0\}$ ,  $J_+ := \{j \mid t_j > 0\}$ .
  - Kopiere alle Spalten von  $T^{\mu-1}$  mit Indizes aus  $J_0$  nach  $T^\mu$ ; das sind die Reaktionen, die das betrachtete Molekül  $i_\mu$  gar nicht verwenden.
  - Von den Reaktionen, die das Molekül verbrauchen (Indizes  $J_-$ ) und solchen, die es erzeugen (Indizes  $J_+$ ), werden nun alle möglichen Kombinationen gebildet, die Molekül  $i_\mu$  ausbalancieren, und dem Tableau  $T^\mu$  als neue Spalten hinzugefügt. So werden aus den  $|J_-| + |J_+|$  alten Spalten  $|J_-| \cdot |J_+|$  neue Spalten. Konkret: Für alle Kombinationen von  $j \in J_+$  und  $j' \in J_-$ , bilde die (positive!) Linearkombination  $|t'_j| \cdot (\text{Spalte } j) + |t_j| \cdot (\text{Spalte } j')$ ; dies erzeugt in Zeile  $i_\mu$  die gewünschte Null.
  - Entferne dabei entstehende redundante Spalten (solche, die sich aus anderen positiv linear kombinieren lassen). Ein Kriterium dafür lautet wie folgt. Sei  $N(j) := \{i : T_{i,j}^\mu = 0\}$  die Menge der Zeilenindizes der Nullen in Spalte  $j$ . Gilt für zwei Spalten  $h \neq j$ , dass  $N(h) \subset N(j)$ , dann kann Spalte  $h$  (weniger Nullen) entfernt werden. Das (zu beweisende) Argument dahinter ist, dass Spalte  $h$  dann aus Spalte  $j$  und weiteren kombiniert werden kann.
4. Nach der letzten Iteration gilt, dass in  $T^M$  alle Zeilen (Moleküle), die an keiner reversiblen externen Reaktion teilnehmen, Nullzeilen sind. Die Spalten entsprechen nun positiven Linearkombinationen von Reaktionen, die die Menge aller bisher betrachteten Moleküle nicht verändern. Da wir im oberen Teil mit der Einheitsmatrix begonnen haben, können wir in dem Teil nun die entsprechenden Koeffizienten der Linearkombinationen ablesen.
5. Die Anzahl der Spalten von  $T^M$  entspricht schon der Anzahl der extreme Pathways. Allerdings müssen wir noch die Moleküle mit Hilfe der reversiblen externen Reaktionen ausbalancieren, die noch nicht bearbeitet wurden. Dies geschieht aber einfach, indem wir für jede bisher nicht bearbeitete Zeile im unteren Teil von  $T^M$  folgendes tun:
- Sei  $t := T_{i,:}^M$  die gerade betrachtete Zeile.
  - Für jede Spalte  $j$  mit  $t_j \neq 0$ , addiere ein geeignetes (ggf. auch negatives) Vielfaches einer Spalte aus  $T^\bullet$  (reversible externe Reaktion; hier gibt es keine Positivitätsbedingung) zu Spalte  $j$ , um  $t_j = 0$  zu erzeugen.
6. Jede Spalte des oberen Teils (die ersten  $n$  Zeilen, ehemals die Identitätsmatrix) von  $T^\mu$  ist nun ein extreme Pathway  $p$ : Diese erfüllen nach Konstruktion  $Sp = 0$  und  $p \geq 0$ . Die Pathways bilden eine nichtnegative  $n \times d$  Pathway-Matrix  $P$ . Da bei der Konstruktion Redundanzen vermieden wurden, lässt sich auch keiner aus den anderen positiv kombinieren.

Tatsächlich wirkt dieser Algorithmus relativ *ad-hoc*. Es sind eine Reihe von Dingen zu beweisen; wir haben versucht, die Wahrheit dieser Aussagen in den Erläuterungen plausibel zu machen.

1. Für alle Spalten  $p$  von  $P$  gilt  $Sp = 0$  und  $p \geq 0$ .
2. Gilt für einen  $n$ -Vektor  $x \geq 0$ , dass  $Sx = 0$ , dann ist  $x = \sum_{k=1}^d \alpha_k p_k$  mit nichtnegativen Koeffizienten  $\alpha_k$ , d.h., jeder nichtnegative stationäre Flux lässt sich als Kombination der extreme Pathways schreiben (nicht notwendig eindeutig).
3. Die Spalten von  $P$  sind konisch unabhängig, d.h., ist  $p_j = \sum_{k=1}^d \alpha_k p_k$ , dann folgt  $\alpha_k = 0$  für  $k \neq j$  und  $\alpha_j = 1$ ; es gibt für die extreme Pathways also keine Darstellung als Kombination anderer extreme Pathways.

**Analyse der Pathway-Matrix.** Mit der  $n \times d$ -Pathway-Matrix  $P$  kann man die gleichen topologischen Analysen wie mit der stöchiometrischen Matrix  $S$  durchführen: Aus  $P$  bilden wir die Inzidenzmatrix  $\hat{P}$ , indem wir alle Nicht-Null-Einträge von  $P$  in  $\hat{P}$  auf Eins setzen. Das  $n \times n$ -Produkt  $\hat{P}\hat{P}^T$  nennen wir *Partizipationsmatrix*; der Eintrag an der Stelle  $(j, l)$  zeigt, an wie vielen extreme Pathways die Reaktionen  $j$  und  $l$  gemeinsam beteiligt sind (partizipieren); die Diagonalelemente  $(j, j)$  geben damit an, an wie vielen extreme Pathways Reaktion  $j$  überhaupt beteiligt ist. Das  $d \times d$ -Produkt  $\hat{P}^T\hat{P}$  nennen wir *Längenmatrix*; der Eintrag an der Stelle  $(q, r)$  zeigt, wie viele Reaktionen die Pathways  $q$  und  $r$  gemeinsam haben (gemeinsame Pfadlänge); die Diagonalelemente  $(q, q)$  geben damit die Länge des Pathways  $q$  an.

Von  $P$  kann man eine Singulärwertzerlegung  $P = U\Sigma V^T$  berechnen und beispielsweise eine Rang-1-Approximation zu  $P$  berechnen. Damit kann man sehen, welche Kombinationen von extreme Pathways für die Konstruktion von  $P$  wichtiger sind als andere.

## 5.8 Abstraktion: Double Description Method

Das grundlegende Problem bei der Berechnung der extreme Pathways kann wie folgt formuliert werden. Gegeben ist eine Punktmenge der Form  $X = \{x \mid Ax \geq 0\}$  mit einer Matrix  $A$ . Gesucht ist eine Darstellung der Form  $X = \{\sum_k \lambda_k p_k \mid \lambda \geq 0\} = \{P\lambda \mid \lambda \geq 0\}$ , und zwar mit einem möglichst kleinen  $P$  (wenige Spalten).

Wir vernachlässigen für die Berechnung von extreme Pathways aus der stöchiometrischen Matrix  $S$  jetzt, dass es reversible externe Reaktionen geben kann, bzw. wir modellieren diese ebenfalls einfach als zwei Einwegreaktionen. Wir können nun

1.  $A := \begin{bmatrix} S \\ -S \\ \text{Id} \end{bmatrix}$  setzen; damit ist offensichtlich  $\{x \mid Ax \geq 0\} = \{x \mid Sx = 0, x \geq 0\}$ , und die  $p_k$  entsprechen direkt den extreme Pathways wie im vorigen Abschnitt;
2. zunächst (beispielsweise mit der SVD von  $S$ ) eine Basis  $B = (b_1, \dots, b_q)$  von  $\text{Kern}(S)$  berechnen; es ist dann automatisch  $Sb_j = 0$  für alle  $j$ . Gesucht sind nun solche Linearkombinationen der  $b_j$ , die nichtnegative Einträge haben, also  $x = \sum_{j=1}^q \mu_j b_j = B \cdot \mu \geq 0$  mit  $\mu \in \mathbb{R}^q$ . Hier ist also die Koeffizientenmenge  $M = \{\mu \mid B\mu \geq 0\}$  gegeben und eine Darstellung  $M = \{R\lambda \mid \lambda \geq 0\}$  gesucht. Damit kann man dann  $BR$  berechnen, um die extreme Pathways zu erhalten.

Wir befassen uns jetzt allgemein mit dem genannten Problem und einem Algorithmus zu seiner Lösung, der *Double Description Method*; wir richten dabei uns nach der Darstellung von Fukuda und Prodon ?.

Der Name kommt daher, dass die Menge  $X$  auf zwei Arten beschrieben wird:

$$X = \{ x \mid Ax \geq 0 \} = \{ P\lambda \mid \lambda \geq 0 \}.$$

Ein Paar von Matrizen  $(A, P)$ , für das diese Beziehung gilt, nennt man ein DD-Paar. Die Spalten von  $P$  nennt man, sofern  $P$  minimal ist, *extreme Strahlen* des Kegels  $X$ . Achtung:  $P$  in einem DD-Paar  $(A, P)$  muss nicht minimal sein, sondern kann Strahlen (Spalten) enthalten, die nicht extrem sind, sich also aus anderen kombinieren lassen. Diese nennen wir *redundante Strahlen*; diese müssen zu einem geeigneten Zeitpunkt identifiziert und entfernt werden.

Ein Satz von Minkowski und ein Satz von Weyl stellen sicher, dass es zu jedem  $A$  ein  $P$  und zu jedem  $P$  ein  $A$  gibt, zu dass  $(A, P)$  ein DD-Paar ist. Ferner gilt:  $(A, P)$  ist genau dann ein DD-Paar, wenn  $(P^T, A^T)$  eins ist (Dualität).

**Algorithmus.** Sei  $A \in \mathbb{R}^{m \times n}$  gegeben mit  $m \geq n$  und vollem  $\text{rang}(A) = n$ . Dies ist in den Anwendungen gegeben. Gesucht ist  $P \in \mathbb{R}^{n \times q}$  mit geeignetem  $q$ .

Die Hauptidee ist, die Matrix  $A$  zeilenweise abzarbeiten. Sei daher  $I \subset \{1, \dots, m\}$  eine Teilmenge der Zeilenindizes und  $A_I$  die entsprechende Teilmatrix von  $A$ .

Angenommen, wir haben bereits eine Matrix  $P$  gefunden, so dass  $(A_I, P)$  ein DD-Paar ist, dann wählen wir eine Zeile  $i \notin I$  aus und konstruieren eine Matrix  $P'$ , so dass  $(A_{I \cup \{i\}}, P')$  ein DD-Paar ist. Sei  $A_i$  die betrachtete  $i$ -te Zeile von  $A$ . Der Raum  $\mathbb{R}^n$  wird durch die neue Ungleichung  $A_i x \geq 0$  in drei Teile geteilt (zwei Halbräume und eine Hyperebene):

$$\begin{aligned} H_i^+ &:= \{ x \in \mathbb{R}^n \mid A_i x > 0 \}, \\ H_i^0 &:= \{ x \in \mathbb{R}^n \mid A_i x = 0 \}, \\ H_i^- &:= \{ x \in \mathbb{R}^n \mid A_i x < 0 \}. \end{aligned}$$

Die Spalten von  $P$  (die die Richtungen der bisherigen Strahlen darstellen) werden dadurch in drei Klassen eingeteilt; dabei sei  $J$  die Indexmenge der Spalten von  $P$  und  $p_j$  die  $j$ -te Spalte von  $P$  für  $j \in J$ :

$$\begin{aligned} J_i^+ &:= \{ j \in J \mid p_j \in H_i^+ \}, \\ J_i^0 &:= \{ j \in J \mid p_j \in H_i^0 \}, \\ J_i^- &:= \{ j \in J \mid p_j \in H_i^- \}, \end{aligned}$$

Die Strahlen mit den Indizes aus  $J_i^+$  und  $J_i^0$  werden übernommen (ihre Richtungen erfüllen ja bereits  $A_i x \geq 0$ ); die Strahlen mit den Indizes aus  $J_i^-$  verletzen aber die Ungleichung. Also werden wir neue (extreme) Strahlen erzeugen, die aus der Kombination jeweils eines Strahls aus  $J_i^+$  und  $J_i^-$  entstehen und entlang  $H_i^0$  verlaufen; dazu bilden wir die Strahlenkombinationen

$$p'_{j,j'} = (A_i p_j) p_{j'} - (A_i p_{j'}) p_j \text{ für } (j, j') \in J_i^+ \times J_i^-.$$

wir ersetzen also  $|J_i^-|$  Strahlen durch  $|J_i^-| \cdot |J_i^+|$  neue Strahlen.

Dabei können allerdings redundante Strahlen entstehen; da wir nur an extremen Strahlen interessiert sind, soll dies verhindert werden.

Tatsächlich genügt es, unter der Annahme, dass  $P$  minimal ist (also nur den extremen Strahlen für  $A_I$  besteht), bestimmte Kombinationen von Strahlen aus  $J_i^+$  und  $J_i^-$  zu bilden, nämlich sogenannte *adjazente* Strahlen, damit auch  $P'$  aus den extremen Strahlen für  $A_{I \cup \{i\}}$  besteht. Es sei  $Z(p) := \{i \mid A_i p = 0\}$ , die Indexmenge aller Zeilen  $i$  von  $A$ , auf denen  $p$  senkrecht steht. Dann heißen zwei extreme Strahlen  $p$  und  $p'$  *adjazent*, wenn gilt: Ist  $p''$  ein Strahl mit  $Z(p) \cap Z(p') \subset Z(p'')$ , dann ist  $p'' = p$  oder  $p'' = p'$  (bis auf positive Skalierung). Sei

$$Adj_i := \{(j, j') \in J_i^+ \times J_i^- \mid p_j \text{ und } p_{j'} \text{ adjazent in } P\}.$$

Dann bilden wir statt aller Strahlenkombinationen nur

$$p'_{j,j'} = (A_i p_j) p_{j'} - (A_i p_{j'}) p_j \text{ für } (j, j') \in Adj_i.$$

Der Beweis des folgenden Satzes, der die Korrektheit des Verfahrens garantiert, findet sich in ?.

**5.8 Satz.** Sei  $A \in \mathbb{R}^{m \times n}$  mit  $m \geq n$ ; sei  $I \subset \{1, \dots, m\}$  und  $i \notin I$ . Sei  $(A_I, P)$  ein DD-Paar mit  $\text{rang}(A_I) = n$ . Dann ist auch  $(A_{I \cup \{i\}}, P')$  ein DD-Paar, wobei  $P'$  eine  $n \times |J'|$ -Matrix ist mit

$$\begin{aligned} J' &:= J_i^+ \cup J_i^0 \cup Adj_i, \\ Adj_i &:= \{(j, j') \in J_i^+ \times J_i^- \mid p_j \text{ und } p_{j'} \text{ adjazent in } P\}, \end{aligned}$$

und mit Spalten  $p'$  wie folgt:

$$\begin{aligned} p'_{j^+} &:= p_{j^+} && \text{für } j^+ \in J_i^+, \\ p'_{j^0} &:= p_{j^0} && \text{für } j^0 \in J_i^0, \\ p'_{j,j'} &= (A_i p_j) p_{j'} - (A_i p_{j'}) p_j && \text{für } (j, j') \in Adj_i. \end{aligned}$$

Ist  $P$  minimal für  $A_I$  (enthält also nur extreme Strahlen), dann ist  $P'$  minimal für  $A_{I \cup \{i\}}$ .

Wir müssen noch beschreiben, wie der Algorithmus beginnt. Natürlich kann man zu einer einzelnen Zeile  $a$  passende Strahlen  $p$  finden so dass  $a \cdot x \geq 0$  (dies ist ein Halbraum) äquivalent ist mit  $x = \sum_j \alpha_j p_j$ ,  $\alpha \geq 0$ : Wir benötigen eine orthogonale Basis für die Hyperebene  $a \cdot x = 0$  und zu jedem Basisvektor den entsprechenden negativen Vektor. Damit können wir jeden Punkt der Hyperebene darstellen. Weiter benötigen wir den Strahl  $a$  selbst; dieser ist ja der Normalenvektor der Hyperebene und steht senkrecht darauf.

Eine effizientere Initialisierung besteht darin, nicht mit einer Zeile, sondern mit  $n$  linear unabhängigen Zeilen zu beginnen (diese existieren, da wir  $\text{rang}(A) = n$  vorausgesetzt hatten). Die entsprechende Submatrix  $A_I$  mit  $|I| = n$  ist dann invertierbar, und  $P = A_I^{-1}$  liefert entsprechende Strahlen, denn ist  $x = A_I^{-1} \lambda$  mit  $\lambda \geq 0$ , dann ist  $A_I x = \lambda \geq 0$  und umgekehrt.

Der Zusammenhang (Bildung von Linearkombinationen von Strahlen) mit der in Abschnitt 5.7 beschriebenen Methode ist offensichtlich. Weniger klar ist die Äquivalenz der Kriterien zur Vermeidung von redundanten Strahlen.

**Komplexität.** Die Komplexität der DD-Methode ist im schlimmsten Fall exponentiell. Es ist möglich, dass in Zwischenschritten viele neue Spalten generiert werden, die nachher als konisch abhängig erkannt und wieder gestrichen werden. Die Laufzeit hängt daher stark davon ab, in welcher Reihenfolge die Zeilen von  $A$  bearbeitet werden. In der Praxis kann man mit guten Heuristiken für die Wahl der Reihenfolge überflüssigen Aufwand vermeiden. Die tatsächliche Komplexität des Problems ist ungeklärt. Analoges gilt für die in Abschnitt 5.7 vorgestellte Methode, die letzten Endes eine angepasste Version der DD-Methode ist.



---

# Matrixalgebra

---

## A.1 Grundlagen

Sei  $A \in \mathbb{R}^{m \times n}$  eine Matrix. Sie wird als lineare Abbildung  $\mathbb{R}^n \rightarrow \mathbb{R}^m$  aufgefasst:

$$\mathbb{R}^n \ni x \mapsto Ax \in \mathbb{R}^m.$$

Dabei ist  $(Ax)_i = \sum_{j=1}^n A_{ij}x_j$  für alle  $i = 1, \dots, m$ .

**A.1 Definition** (Bild, Kern). Es ist  $\text{Bild}(A) := \{Ax : x \in \mathbb{R}^n\} \subset \mathbb{R}^m$  die Menge aller Bilder, die von Vektoren  $x \in \mathbb{R}^n$  erzeugt werden können. Dies entspricht der Menge aller Linearkombinationen der Spalten von  $A$  (das *Bild* von  $A$ )

Es ist  $\text{Kern}(A) := \{x : Ax = 0\} \subset \mathbb{R}^n$  die Menge aller Vektoren, die auf die Null abgebildet werden (der *Kern* von  $A$ ).

**A.2 Definition** (Spur). Die *Spur* einer quadratischen Matrix ist die Summe ihrer Diagonalelemente,

$$\text{Spur}(A) = \sum_{i=1}^n A_{ii}.$$

**A.3 Definition** (Transponierte Matrix, Symmetrie). Für  $A \in \mathbb{R}^{m \times n}$  ist die *transponierte* Matrix  $A^T \in \mathbb{R}^{n \times m}$  definiert durch

$$A_{ji}^T := A_{ij}.$$

Eine Matrix  $A$  heißt *symmetrisch*, wenn sie quadratisch ist ( $m = n$ ) und  $A = A^T$  ist.

**A.4 Definition** (positiv (semi)definit). Sei  $S \in \mathbb{R}^{n \times n}$  symmetrisch. Dann heißt  $S$  *positiv semidefinit*, wenn  $x^T S x \geq 0$  für alle  $x \in \mathbb{R}^n$ , und *positiv definit*, wenn  $x^T S x > 0$  für alle  $x \neq 0$ .

**A.5 Definition** (Skalarprodukt, Orthogonalität). Sind  $u, v \in \mathbb{R}^n$ , dann ist ihr *Skalarprodukt* (*inneres Produkt*) definiert als Summe der komponentenweisen Produkte, bzw.

$$u^T \cdot v = \sum_{i=1}^n u_i v_i.$$

Ist  $u^T v = 0$ , nennt man  $u$  und  $v$  *orthogonal*.

**A.6 Definition** (Frobeniusnorm). Die Frobeniusnorm  $\|A\|_F$  einer Matrix ist die Wurzel Summe der Quadrate ihrer Einträge. Es ist

$$\|A\|_F^2 = \sum_{i,j} A_{ij}^2 = \text{Spur}(A^T A).$$

Der Frobeniusabstand zweier Matrizen  $A, B$  der gleichen Größe ist

$$d_F(A, B) := \|A - B\|_F, \quad \text{also} \\ d_F^2(A, B) = \sum_{i,j} (A_{ij} - B_{ij})^2 = \text{Spur}((A - B)^T (A - B)).$$

Wichtig sind folgende Fakten aus der linearen Algebra, deren Beweise gute Übungsaufgaben sind.

- $\text{Kern}(A)$  ist ein Unterraum des  $\mathbb{R}^n$ .
- $\text{Bild}(A)$  ist ein Unterraum des  $\mathbb{R}^m$ . Seine Dimension  $\text{rang}(A)$  heißt *Rang* von  $A$  (genauer: *Spaltenrang*).
- $\text{Bild}(A^T)$  ist ein Unterraum des  $\mathbb{R}^n$ . Seine Dimension heißt *Zeilenrang* von  $A$  und stimmt mit dem Spaltenrang überein.
- $\text{Kern}(A^T)$  ist ein Unterraum des  $\mathbb{R}^m$ .
- Es gilt die Dimensionsformel  $\text{rang}(A) + \dim \text{Kern}(A) = m$ .
- $\text{Kern}(A)$  und  $\text{Bild}(A^T)$  sind orthogonale Komplemente des  $\mathbb{R}^n$ , d.h., ist  $u \in \text{Kern}(A)$  und  $v \in \text{Bild}(A^T)$ , dann ist  $u \perp v$ , d.h.,  $u^T v = 0$ , und jeder Vektor  $x \in \mathbb{R}^n$  lässt sich eindeutig zerlegen in  $x = u + v$  mit  $u \in \text{Kern}(A)$  und  $v \in \text{Bild}(A^T)$ .
- Analog gilt:  $\text{Kern}(A^T)$  und  $\text{Bild}(A)$  sind orthogonale Komplemente des  $\mathbb{R}^m$ .
- Die Matrix  $A^T A$  ist  $n \times n$ , symmetrisch und positiv semidefinit. Analog ist  $AA^T$  eine  $m \times m$ -Matrix, symmetrisch und positiv semidefinit.
- Es ist  $\text{Kern}(A^T A) = \text{Kern}(A)$  und  $\text{Kern}(AA^T) = \text{Kern}(A^T)$ .

## A.2 Eigenwerte

**A.7 Definition** (Eigenwert, Eigenvektor). Sei  $A \in \mathbb{R}^n \times n$  eine quadratische Matrix. Wenn es eine Zahl  $\lambda \in \mathbb{R}$  und ein  $x \in \mathbb{R}^n$ ,  $x \neq 0$  gibt mit

$$Ax = \lambda x,$$

dann heißt  $x$  *Eigenvektor* von  $A$  zum *Eigenwert*  $\lambda$ .

Auf Eigenvektoren wirkt  $A$  also wie eine Streckung oder Stauchung (die Richtung  $x$  wird um den Faktor  $|\lambda| > 1$  gestreckt bzw. um den Faktor  $0 \leq |\lambda| < 1$  gestaucht), ggf. verbunden mit einer Spiegelung am Nullpunkt (wenn  $\lambda < 0$ ).

Besonders erstrebenswert ist es, wenn es eine Basis  $(b_1, \dots, b_n)$  des  $\mathbb{R}^n$  aus Eigenvektoren  $\lambda_1, \dots, \lambda_n$  gibt (die gibt es aber nicht immer!), denn dann kann man jeden Vektor  $x$  eindeutig als Linearkombination der Eigenvektoren schreiben und  $A$  einfach auf die Eigenvektoren anwenden:

$$x = \sum_i \beta_i b_i \quad \implies \quad Ax = \sum_i \beta_i \lambda_i b_i.$$

Wie kann man herausfinden, welche  $\lambda \in \mathbb{R}$  Eigenwerte sind? Nun,  $Ax = \lambda x$  ist äquivalent zu  $(A - \lambda \text{Id})x = 0$ ; es gibt also  $x \neq 0$ , die dieses homogene Gleichungssystem erfüllen, d.h.  $A - \lambda \text{Id}$  hat nicht vollen Rang. Das wiederum ist gleichbedeutend damit, dass die Determinante verschwindet:

$$\lambda \text{ Eigenwert} \iff \det(A - \lambda \text{Id}) = 0.$$

Nach dem Determinantenentwicklungssatz ist  $\det(A - \lambda \text{Id})$ , wenn man  $\lambda$  als variable betrachtet ein Polynom vom Grad  $n$  in  $\lambda$ ; es wird das *charakteristische Polynom* zu  $A$  genannt. Dies hat genau  $n$  (komplexe, nicht notwendig verschiedene) Nullstellen; dies sind die Eigenwerte. Dabei wird jede Nullstelle  $\lambda_0$  so oft gezählt, wie  $(\lambda_0 - \lambda)$  Faktor des Polynoms ist. Dies nennt man die *algebraische Vielfachheit* des Eigenwerts  $\lambda_0$ .

Hat man ein  $\lambda_0$  als Eigenwert erkannt, sucht man die zugehörigen Eigenvektoren  $x$ , indem man den Lösungsraum von  $(A - \lambda_0 \text{Id})x = 0$  bestimmt, also eine Basis von  $\text{Kern}(A - \lambda_0 \text{Id})$ . Hierbei kann man eine unangenehme Überraschung erleben: Die Dimension des Kerns kann echt kleiner sein als die Vielfachheit der Nullstelle; die Dimension des Kerns heißt die *geometrische Vielfachheit* des Eigenwerts.

Es gibt also zwei Gründe, warum es normalerweise keine (reelle) Basis aus Eigenvektoren gibt: komplexe Nullstellen des charakteristischen Polynoms (dieses Problem verschwindet im Komplexen) und dass die geometrische Vielfachheit eines Eigenwerts kleiner ist als seine algebraische Vielfachheit. Die Beispiele

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \text{ (Drehung um 90 Grad)} \quad \text{und} \quad \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \text{ (Scherung)}$$

illustrieren die beiden Probleme mit jeweils einem Minimalbeispiel.

Wenn man Glück hatte und eine Basis  $B = (b_1, \dots, b_n)$  aus Eigenvektoren gefunden hat, schreibt man diese als Spalten in eine Matrix  $B$ ; auf jede Spalte  $b_i$  wirkt  $A$  dann einfach als Multiplikation mit  $\lambda_i$ .

$$Ab_i = \lambda_i b_i \quad \text{oder} \quad AB = B\Lambda$$

mit  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ . Da  $B$  aus einer Basis besteht, also vollen Rang hat, ist  $B$  invertierbar, und

$$\Lambda = B^{-1}AB.$$

Man sagt hierfür auch,  $A$  wird mit Hilfe der Basis  $B$  diagonalisiert.

Eine spezielle Klasse von Matrizen, für die dies möglich ist, sind symmetrische Matrizen. Um dies einzusehen, müssen wir zunächst über orthogonale Matrizen sprechen.

## A.3 Orthogonale Matrizen

**A.8 Definition.** Eine Matrix  $U \in \mathbb{R}^{n \times n}$  aus Spalten  $u_1, \dots, u_n$  heißt orthogonal, wenn

$$u_i^T u_j = \delta_{ij} := \begin{cases} 1 & \text{wenn } i = j, \\ 0 & \text{sonst,} \end{cases}$$

wenn also  $U^T U = \text{Id}$ . Die Spalten einer orthogonalen Matrix bilden ein *Orthonormalsystem*: Sie sind orthogonal zueinander und (auf Länge 1) normalisiert. Sie bilden auch eine Basis; eine *Orthonormalbasis*. Eine orthogonale Matrix ist also invertierbar; die Inverse ist die Transponierte:  $U^{-1} = U^T$ .

Hat man  $r \leq n$  zueinander orthogonale Vektoren  $u_1, \dots, u_r$ , so kann man durch Skalierung auf die (euklidische) Länge 1 erreichen, dass diese für sich ein Orthonormalsystem bilden.

Nach dem Basisergänzungssatz kann dieses zu einer Orthonormalbasis  $u_1, \dots, u_n$  im  $\mathbb{R}^n$  ergänzt werden. Kozpetionell wählt man zufällig einen neuen Vektor  $u \in \mathbb{R}^n$  und berechnet die Projektion  $u'$  auf den von den existierenden  $u_i$  aufgespannten Unterraum. Dann steht  $u - u'$  darauf senkrecht und bildet nach Normierung einen weiteren Basisvektor. Diese Prozedur wiederholt man, bis  $n$  orthonormale Vektoren vorliegen. Tatsächlich ist wegen der Differenzbildung  $u - u'$  diese Prozedur numerisch nicht zu empfehlen, wenn manche Komponenten von  $u$  und  $u'$  in etwa gleich groß sind (Auslöschung).

**A.9 Lemma** (Invarianz der Frobeniusnorm und euklidischen Norm unter orthogonalen Transformationen). *Die Multiplikation einer Matrix  $A$  mit einer orthogonalen Matrix (von links oder rechts) lässt die Frobeniusnorm von  $A$  unverändert: Sei  $A \in \mathbb{R}^{m \times n}$  beliebig; seien  $U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}$  jeweils orthogonal. Dann ist*

$$\|A\|_F = \|UA\|_F = \|AV\|_F.$$

Analog gilt: *Multiplikation mit einer orthogonalen Matrix lässt die euklidische Norm eines Vektors  $v \in \mathbb{R}^m$  unverändert:*

$$\|Uv\|_2 = \|v\|_2.$$

*Anschaulich: Orthogonale Transformationen sind längenerhaltend.*

**Beweis.** Zunächst ist  $\|A\|_F^2 = \text{Spur}(A^T A)$ . Damit ist

$$\|UA\|_F^2 = \text{Spur}((UA)^T (UA)) = \text{Spur}(A^T U^T U A) = \text{Spur}(A^T A) = \|A\|_F^2.$$

Analog ist  $\|Uv\|_2^2 = (Uv)^T (Uv) = v^T U^T U v = v^T v = \|v\|_2^2$ . □

Nun zu dem wichtigen Resultat, das besagt, dass sich eine symmetrische Matrix stets diagonalisieren lässt, und dies sogar mit einer Orthonormalbasis.

**A.10 Satz.** *Eine symmetrische Matrix  $S$  lässt sich mit einer orthogonalen Matrix diagonalisieren, d.h., es gibt eine orthohonale Matrix  $U$  der gleichen Größe wie  $S$  mit  $U^T U = U U^T = \text{Id}$  und  $U^T S U$  diagonal.*

**Beweis.** Man zeigt zwei Dinge; durch iterative Anwendung folgt dann das Lemma insgesamt.

1. Zu einer symmetrischen Matrix  $S$  existiert ein reeller Eigenwert  $\lambda$  mit zugehörigem Eigenvektor  $x$ , also mit  $Sx = \lambda x$ . (Allgemein gilt dies nicht, es könnten beispielsweise alle Eigenwerte komplex sein.)
2. Ist  $x$  ein Eigenvektor von  $S$  und  $y \perp x$ , dann ist auch  $Sy \perp x$ . Mit anderen Worten:  $S$  bildet das orthogonale Komplement von  $x$  auf sich ab. Beweis:  $(Sy)^T x = y^T S^T x = y^T Sx = \lambda y^T x = \lambda \cdot 0 = 0$ .

Wegen 1. existiert zunächst ein Eigenvektor. Wegen 2. operiert  $S$  unabhängig auf  $x$  und seinem orthogonalen Komplement  $Y = \{y \in \mathbb{R}^n \mid x^T y = 0\}$ . Wir nehmen also  $x$  als ersten Basisvektor in  $U$  auf und betrachten danach  $S$  nur noch eingeschränkt auf das orthogonale Komplement von  $x$ ; dort finden wir wiederum einen Eigenvektor, u.s.w.  $\square$

Aus dem Satz folgt insbesondere: Ist  $S$  symmetrisch, dann sind äquivalent:

$$\begin{aligned} S \text{ ist positiv semidefinit} &\iff \text{Alle } n \text{ Eigenwerte von } S \text{ sind nichtnegativ.} \\ S \text{ ist positiv definit} &\iff \text{Alle } n \text{ Eigenwerte von } S \text{ sind positiv.} \end{aligned}$$

## A.4 Die Singulärwertzerlegung

Es sei  $A \in \mathbb{R}^{m \times n}$ , oBdA mit  $m \geq n$ . (Andernfalls transponiere man  $A$ ; der Satz gilt genauso.) Ist  $A$  die Koeffizientenmatrix eines linearen Gleichungssystems  $Ax = b$  mit  $b \in \mathbb{R}^m$  bekannt und  $x \in \mathbb{R}^n$  unbekannt, so gebe es also mehr Gleichungen als Unbekannte.

**A.11 Satz** (Singulärwertzerlegung). *Sei  $A \in \mathbb{R}^{m \times n}$  mit  $r := \text{rang}(A) \leq \min\{m, n\}$ . Dann existieren orthogonale Matrizen  $U = (u_1, \dots, u_m) \in \mathbb{R}^{m \times m}$  und  $V = (v_1, \dots, v_n) \in \mathbb{R}^{n \times n}$  derart, dass*

$$U^T A V = \text{diag}(\sigma_1, \dots, \sigma_r) =: \Sigma \in \mathbb{R}^{m \times n};$$

*dies ist eine  $m \times n$ -Matrix mit  $r$  von Null verschiedenen Elementen auf der Hauptdiagonalen, mit  $\sigma_1 \geq \dots \geq \sigma_r > 0$ .*

*Dabei sind  $\sigma_1^2, \dots, \sigma_r^2$  die (gemeinsamen) positiven Eigenwerte von  $AA^T$  und  $A^T A$ . Insbesondere sind die  $\sigma_i$  eindeutig durch  $A$  bestimmt und heißen singuläre Werte von  $A$ .*

Zum Beweis benötigen wir das folgende Lemma, das hier nicht bewiesen wird

**A.12 Lemma.** *Die Matrizen  $AA^T \in \mathbb{R}^{m \times m}$  und  $A^T A \in \mathbb{R}^{n \times n}$  haben den gleichen Rang  $r \leq n$ ; ihre positiven (d.h., von Null verschiedenen) Eigenwerte sind gleich und haben die gleiche Vielfachheit. Ferner ist  $\text{Kern}(A) = \text{Kern}(A^T A)$  und  $\text{Kern}(A^T) = \text{Kern}(AA^T)$ .*

**Beweis.** [Satz A.11] Seien  $\lambda_1 \geq \dots \geq \lambda_r > 0 = \lambda_{r+1} = \dots = \lambda_n$  die Eigenwerte von  $A^T A$  und  $(v_1, \dots, v_n)$  ein zugehöriges Orthonormalssystem von Eigenvektoren, d.h.,  $V^T(A^T A)V = \Lambda$ .

Für  $1 \leq i \leq r$  setze  $\sigma_i := \sqrt{\lambda_i}$  und  $u_i := \frac{1}{\sigma_i} A v_i$ .

Wir behaupten:  $(u_i)_{i=1,\dots,r}$  ist ein Orthonormalsystem von Eigenvektoren zu  $AA^T$ . Beweis: Für  $1 \leq i, j \leq r$  ist  $u_i^T u_j = \frac{1}{\sigma_i \sigma_j} v_i^T A^T A v_j = \frac{1}{\sigma_i \sigma_j} (V^T (A^T A) V)_{ij} = \delta_{ij}$ . Dieses Orthonormalsystem kann (beliebig) zu einer Orthonormalbasis der Größe  $m$  ergänzt werden. (Man sieht hierbei schon: Die genaue Wahl der  $u_{r+1}, \dots, u_m$  spielt dabei keine Rolle.)

Seien nun  $U, V$  wie im Satz angegebenen Matrizen aus diesen Orthonormalsystemen. Für  $1 \leq i, j \leq r$  ist

$$(U^T A V)_{ij} = u_i^T A v_j = \frac{1}{\sigma_i} v_i^T A^T A v_j = \sigma_i \delta_{ij}.$$

Ist  $j > r$ , dann ist  $A^T A v_j = 0$ , wegen  $\text{Kern}(A^T A) = \text{Kern}(A)$  auch  $A v_j = 0$ , also auch  $(U^T A V)_{ij} = u_i^T A v_j = 0$ . Analog ist für  $i > r$  ebenfalls  $(U^T A V)_{ij} = u_i^T A v_j = 0$ .  $\square$

Einige Bemerkungen:

- Eine Singulärwertzerlegung (singular value decomposition, SVD) ist nicht notwendig eindeutig; die Wahl der "auffüllenden"  $u_i$  im Beweis ist beispielsweise eine Quelle für Beliebigkeit. Wir sprechen also allgemein von *einer* Singulärwertzerlegung, nicht von *der* Singulärwertzerlegung. Die Singulärwerte (und ihre Vielfachheiten) sind aber eindeutig bestimmt, sowie auch der zu einem Singulärwert gehörende aufgespannte Raum der  $u_i$  und  $v_i$ . Hat ein solcher die Dimension 1, so sind auch  $u_i$  und  $v_i$  dank Normierung eindeutig festgelegt.
- Eine SVD einer Matrix  $A$  liefert orthonormale Basen der vier zu  $A$  assoziierten Unterräume. Sei

$$A = U \Sigma V^T$$

eine Singulärwertzerlegung von  $A \in \mathbb{R}^{m \times n}$  mit  $U = (u_1, \dots, u_m) \in \mathbb{R}^{m \times m}$ ,  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{m \times n}$  vom Rang  $r$  und  $V = (v_1, \dots, v_n) \in \mathbb{R}^{n \times n}$ . Ferner sei  $\sigma_i = 0$  für  $i > r$ . Dann ist

$$\begin{aligned} A v_j &= \sigma_j u_j, \\ A^T u_i &= \sigma_i v_i. \end{aligned}$$

Damit ist klar:

1. Eine (orthonormale) Basis für  $\text{Kern}(A)$  ist  $\{v_{r+1}, \dots, v_n\}$ .
2. Eine (orthonormale) Basis für  $\text{Bild}(A)$  ist  $\{u_1, \dots, u_r\}$ .
3. Eine (orthonormale) Basis für  $\text{Kern}(A^T)$  ist  $\{u_{r+1}, \dots, u_m\}$ .
4. Eine (orthonormale) Basis für  $\text{Bild}(A^T)$  ist  $\{v_1, \dots, v_r\}$ .

Damit sieht man abermals, dass  $\text{Kern}(A)$  das orthogonale Komplement von  $\text{Bild}(A^T)$  ist.

## A.5 Anwendungen der Singulärwertzerlegung

### A.5.1 Least-Squares-Lösung überbestimmter Gleichungssysteme

Es sei  $Ax = b$  mit  $A \in \mathbb{R}^{m \times n}$ ,  $m \geq n$  zu "lösen". Da man wegen  $m \geq n$  nicht erwarten kann, dass das Gleichungssystem exakt lösbar ist, formuliert man die Aufgabe dahingehend

um, ein (oder alle)  $x$  zu finden, so dass  $\|Ax - b\|_2^2 = \sum_{i=1}^m ((Ax)_i - b_i)^2$  (die Summe der quadrierten Fehler) minimal wird.

Sei  $A = U\Sigma V^T$  eine SVD von  $A$  mit  $U = (u_1, \dots, u_m)$  und  $\text{rang}(A) = r$ . Dann ist

$$\begin{aligned} \|Ax - b\|_2^2 &= \|U\Sigma V^T x - b\|_2^2 \\ &= \|\Sigma V^T x - U^T b\|_2^2 \\ &= \sum_{i=1}^m [\sigma_i (V^T x)_i - u_i^T b]^2 \\ &= \sum_{i=1}^r [\sigma_i (V^T x)_i - u_i^T b]^2 + \sum_{i=r+1}^m (u_i^T b)^2, \end{aligned}$$

da  $\sigma_i = 0$  für  $i > r$ . Der rechte Summand ist ein nicht zu vermeidender Fehler, da er nicht von  $x$  abhängt. Der linke Summand verschwindet genau dann, wenn

$$V^T x = (u_1^T b / \sigma_1, \dots, u_r^T b / \sigma_r, \alpha_{r+1}, \dots, \alpha_n)^T$$

mit beliebigen  $n - r$  Werten  $\alpha_{r+1}, \dots, \alpha_n$  (diese kommen im Fehlerquadrat nicht vor). Sucht man nach der Lösung  $x$  mit minimaler euklidischer Norm  $\|x\|_2$ , dann setzt man wegen  $\|x\|_2 = \|V^T x\|_2$  alle  $\alpha_j = 0$  und erhält nach Multiplikation mit  $V$  von links die eindeutige Lösung des linearen Ausgleichsproblems mit minimaler euklidischer Norm

$$x^* = V \cdot (u_1^T b / \sigma_1, \dots, u_r^T b / \sigma_r, 0, \dots, 0)^T = \sum_{j=1}^r \frac{u_j^T b}{\sigma_j} v_j.$$

### A.5.2 Die Pseudoinverse einer Matrix

Sei  $A \in \mathbb{R}^{m \times n}$  mit Rang  $r$ . Sei  $A = U\Sigma V^T$  eine SVD mit  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{m \times n}$ . Sei  $\Sigma^\dagger := \text{diag}(1/\sigma_1, \dots, 1/\sigma_r) \in \mathbb{R}^{n \times m}$ . Setze

$$A^\dagger := V\Sigma^\dagger U^T \in \mathbb{R}^{n \times m}.$$

Dann ist  $AA^\dagger = \text{Id}_{r,m}$  und  $A^\dagger A = \text{Id}_{r,n}$ ; dabei ist  $\text{Id}_{r,m}$  eine  $m \times m$ -Nullmatrix mit  $r$  Einsen auf der Hauptdiagonalen.

Ist  $r = m = n$ , also  $A$  invertierbar, folgt  $A^\dagger = A^{-1}$ . Man kann daher  $A^\dagger$  als verallgemeinerte Inverse (für rang-defizite und für rechteckige Matrizen) auffassen. (Dies ist nicht die einzige Möglichkeit, die Inverse zu verallgemeinern.)

Eine berechnete Frage ist, ob  $A^\dagger$  überhaupt wohldefiniert ist, denn wir sind ja von einer (beliebigen) SVD von  $A$  ausgegangen, um  $A^\dagger$  zu definieren, und die SVD ist nicht eindeutig. Allerdings gilt folgender Satz, der besagt, dass man mit jedem so definierten  $A^\dagger$  die (eindeutige) Lösung des linearen Ausgleichsproblems aus dem vorigen Abschnitt für alle rechten Seiten  $b \in \mathbb{R}^m$  berechnen kann; damit ist  $A^\dagger : \mathbb{R}^m \rightarrow \mathbb{R}^n$  also wohldefiniert.

**A.13 Satz.** Seien  $A \in \mathbb{R}^{m \times n}$  und  $b \in \mathbb{R}^m$  gegeben. Sei  $x^* \in \mathbb{R}^n$  die (eindeutige) Lösung minimaler Länge des linearen Ausgleichsproblems

$$\text{Minimiere } \|Ax - b\|_2.$$

Dann gilt

$$x^* = A^\dagger \cdot b.$$

**Beweis.** Einfach nachrechnen:  $A^\dagger b = V\Sigma^\dagger U^T b$ ; das ist identisch mit dem Ausdruck für  $x^*$  im vorigen Abschnitt.  $\square$

Ferner kann man zeigen: Ist  $m \geq n$  und hat  $A$  vollen Rang  $n$ , dann ist  $A^T A$  invertierbar und es gilt

$$A^\dagger = (A^T A)^{-1} A^T.$$

Ansonsten ist  $A^T A + \delta \text{Id}_n$  für hinreichend kleine  $\delta > 0$  invertierbar und es ist

$$A^\dagger = \lim_{\delta \rightarrow 0} (A^T A + \delta \text{Id}_n)^{-1} A^T.$$

### A.5.3 Die beste Rang- $k$ -Approximation einer Matrix

Aus  $A = U\Sigma V^T$  mit Rang  $r$  folgt die Darstellung von  $A$  als Summe von Rang-1-Matrizen:

$$A = \sum_{i=1}^r \sigma_i u_i v_i^T;$$

dabei ist jedes  $u_i v_i^T$  eine  $m \times n$ -Matrix mit Rang 1 (“äußeres Produkt”).

Gesucht ist nun eine Matrix  $A_k$  mit Rang  $k \leq r$ , die  $A$  (unter allen Rang- $k$ -Matrizen) am besten approximiert. Es ist naheliegend, die ersten  $k$  Terme der SVD-Zerlegung zu benutzen, da diese den größten Singulärwerten  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k$  entsprechen, also

$$A_k = \sum_{i=1}^k \sigma_i u_i v_i^T = U \cdot \text{diag}(\sigma_1, \dots, \sigma_k) \cdot V^T.$$

Ist dabei  $\sigma_k = \sigma_{k+1}$ , so ist die Lösung nicht eindeutig.

Dies kann man beispielsweise zur Bildkompression benutzen: Ein (Graustufenbild) wird als  $A \in [0, 1]^{m \times n}$  dargestellt (0 bedeute schwarz, 1 bedeute weiß); diese Matrix wird durch ihre beste Rang- $k$ -Approximation für  $k \ll \min\{m, n\}$  ersetzt. Man muss nun nur jeweils die  $k$  nötigen  $u_i$  und  $v_i$  speichern also  $k(m+n)$  statt  $mn$  Einträge.

Wir wollen zeigen, dass die “abgeschnittene Singulärwertzerlegung” die richtige Wahl zur Rang- $k$ -Approximation ist, wenn man die Frobeniusnorm als Abstandsmaß zwischen zwei Matrizen nimmt. Erinnerung:

$$d_{\mathbb{F}}^2(A, B) = \|A - B\|_{\mathbb{F}}^2 := \sum_{i,j} (A_{ij} - B_{ij})^2.$$

**TODO:**

---

## Literaturverzeichnis

---