Zusatzmaterial

Präferenzmodelle in der Praxis
Analyse von Paarvergleichen, Likert Items
und Rankings mit **R**-prefmod

Grand, Dittrich, Hatzinger
ISBN: 978-3-8252-3785-1

Ad Abschnitt 4.1, Seite 18
Das logistische Bradley-Terry Modell

Das logistische Bradley-Terry (BT) Modell kann durch folgende Charakteristika überblicksmäßig beschrieben werden

- es gibt zwei Antwortkategorien bzw. Entscheidungsmöglichkeiten (z.B. bevorzugt/nicht bevorzugt),
- es wird angenommen, dass die Entscheidungen der UrteilerInnen unabhängig sind, d.h. die Antwort in einem bestimmten Vergleich beeinflusst nicht die Antwort in einem anderen Vergleich,
- die Anzahl der Bevorzugungen in einem Vergleich folgen einer Binomialverteilung (siehe auch Abschnitt 5.1.2),
- das BT Modell ist ein generalisiertes lineares Modell (GLM), siehe Abschnitt 5.1.2 und für Details Aitkin et al. (2009).

Die Formel des BT Modells

$$p\left(Y_{jk} = 1 \,|\, \pi_j, \pi_k\right) = \frac{\pi_j}{\pi_j + \pi_k}\,, \tag{1}$$

lässt sich in exponentierter Form auch folgendermaßen anschreiben

$$p(y_{jk(j)}) = \frac{\exp(\lambda_j)}{\exp(\lambda_j) + \exp(\lambda_k)}\,.$$

Die logarithmierten Objektparameter λ stehen zu den Werteparametern aus Formel (1) in folgender Beziehung: $\ln \pi = \lambda$ bzw. $\pi = \exp(\lambda)$.
Durch Umformung kann man diese Modellformel auch anders darstellen. Man modelliert dann nicht die Wahrscheinlichkeit für eine bestimmte Entscheidung sondern die Chance, bzw. die logarithmierte Chance, für die Wahl einer bestimmten Alternative.
Die logarithmierte Chance, dass im Paarvergleich (jk) Objekt j bevorzugt wird und Objekt k nicht, ergibt sich aus der Division der Wahrscheinlichkeit $p(y_{jk(j)})$, dass Objekt j bevorzugt wird, durch die Gegenwahrscheinlichkeit $p(y_{jk(k)})$, nämlich, dass Objekt k im Vergleich (jk) bevorzugt wird:

$$\operatorname{logit} p(y_{jk(j)}) = \ln\left(\frac{p(y_{jk(j)})}{p(y_{jk(k)})}\right) = \lambda_j - \lambda_k\,. \tag{2}$$

Die Division der Wahrscheinlichkeit durch die Gegenwahrscheinlichkeit bezeichnet man als Chance (odds). In Formel (2) wird diese Chance logarithmiert. Logarithmierte Chancen werden auch als *Logits* bezeichnet. Die logarithmierte Chance, dass Objekt j Objekt k vorgezogen wird, kann auch als eine Funktion der Differenzen zwischen den Objektparametern λ_j und λ_k im jeweiligen Paarvergleich dargestellt werden.
Bei beispielsweise drei Objekten ($J = 3$), bezeichnet mit 1,2 und 3, und daher drei Vergleichen lauten die Gleichungen des logistischen BT Modells

$$\begin{aligned} \operatorname{logit} p(y_{12(1)}) &= \lambda_1 - \lambda_2\,, \\ \operatorname{logit} p(y_{13(1)}) &= \lambda_1 - \lambda_3\,, \\ \operatorname{logit} p(y_{23(2)}) &= \lambda_2 - \lambda_3\,. \end{aligned}$$

Ad Abschnitt 5.1.5, 28
X^2 und Devianz für das LLBT Basismodell

Die X^2 – Statistik

$$X^2 = \sum \frac{(n_{jk(j)} - \hat{m}_{jk(j)})^2}{\hat{m}_{jk(j)}}$$

Die quadrierte Abweichung zwischen beobachteter $n_{jk(j)}$ und erwarteter Häufigkeit $\hat{m}_{jk(j)}$, im Verhältnis zur Größe der erwarteten Häufigkeit $\hat{m}_{jk(j)}$ wird summiert und stellt die Pearson X-Quadrat Statistik dar. Die Summe wird dabei über die beiden Antwortmöglichkeiten $jk(j)$ und $jk(k)$ in allen Paarvergleichen (jk) gebildet.
Je größer der X^2-Wert, desto größer ist die Differenz zwischen den beobachteten und den erwarteten Häufigkeiten und desto „schlechter“ beschreibt das Modell die beobachteten Daten.

Die Devianz

$$\mathrm{D} = 2 \sum n_{jk(j)} \ln\left(\frac{n_{jk(j)}}{\hat{m}_{jk(j)}}\right)$$

Die beobachtete Häufigkeit $n_{jk(j)}$ wird mit dem logarithmierten Quotienten der beobachteten $n_{jk(j)}$ und erwarteten Häufigkeit $\hat{m}_{jk(j)}$ multipliziert und summiert. Die Summe wird dabei wieder über die beiden Antwortmöglichkeiten $jk(j)$ und $jk(k)$ in allen Paarvergleichen (jk) gebildet. Multipliziert man

die gebildete Summe mit 2, erhält man die Devianz. Die Devianz kann auch mit G^2 bezeichnet werden.

Wie die Pearson X^2-Statistik ist auch die Devianz approximativ χ^2-verteilt (siehe z.B. Agresti, 2010; 2013).

Ad Abschnitt 5.1.5, Seite 28
Die Chi-Quadrat Verteilung, p-Wert

Die Chi-Quadrat Verteilung (χ^2) ist eine (theoretische) Wahrscheinlichkeitsverteilung der X^2-Werte (bzw. der Devianz Statistiken), die aus einer unendlichen Zahl von durchgeführten Stichproben aus der selben Grundgesamtheit resultiert. Die Form dieser Verteilung ist von einem Parameter, den sogenannten Freiheitsgraden (df) abhängig und ist in Abbildung 1 mit drei Freiheitsgraden dargestellt.

i *Die Verteilung wird als „theoretisch" bezeichnet, weil man niemals eine unendliche Anzahl von Stichproben erheben kann.*

Nehmen wir beispielsweise den berechneten X^2-Wert oder die Devianz der beobachteten Daten der (fiktiven) Schokoladenstudie aus Abschnitt 5.2 her (siehe R-Output auf Seite 38). Der X^2-Wert für diese Stichprobe beträgt 2.450 und die Devianz 2.438. Für diese beiden Werte der Teststatistiken kann jeweils die Wahrscheinlichkeit (in Form des p-Wertes) berechnet werden, diesen Wert oder einen größeren Wert (bei Geltung des Modells, d.h. Geltung der Nullhypothese) zu beobachten. Zur Berechnung des p-Wertes wird der Wert der Teststatistik und die Anzahl der Freiheitsgrade benötigt.
Ein LLBT-Modell beispielsweise, beschreibt die Daten hinreichend gut, wenn die beobachteten Häufigkeiten $n_{jk(j)}$ in etwa den, unter dem Modell, erwarteten Häufigkeiten $\hat{m}_{jk(j)}$ entsprechen und somit die X^2-Teststatistik (oder die Devianz) klein und der p-Wert groß ist. Nun gilt es zu entscheiden, ab welchem p-Wert man annimmt, dass das Modell die beobachteten Daten hinreichend gut beschreibt (Nullhypothese) und ab welchem p-Wert nicht mehr (Alternativhypothese).

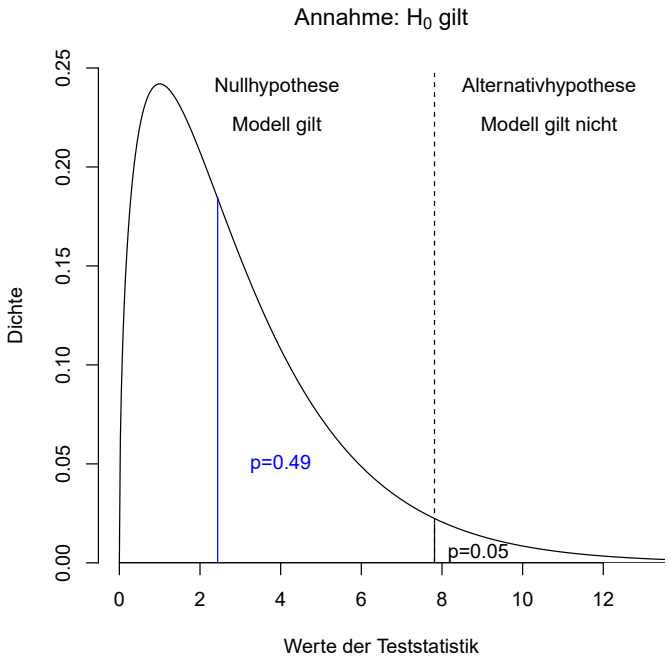

Abbildung 1.: χ^2-Verteilung mit drei Freiheitsgraden

Ist der p-Wert klein (üblicherweise z.B. kleiner als 0.05 oder $p \cdot 100 = 5\%$), dann spricht dies gegen das zugrundliegende Modell. Erst recht gilt dies wenn der der p-Wert kleiner als 0.01 (1%) wäre. Als Entscheidungskriterium wählen wir einen p-Wert von 5%. Ein Teststatistikwert von 7.815 bei 3 Freiheitsgraden entspricht einem p-Wert von 0.05 (5%). Ist $p < 0.05$ würde der Wert der Teststatistik größer als 7.815 sein, sodass er in den rechten Bereich nach der senkrechten Linie fällt, den Bereich der Ablehnung der Nullhypothese, also dass das Modell nicht gilt (siehe Abb. 1).
Ist der p-Wert groß (z.B. 5% oder größer), dann spricht dies mehr oder weniger für das zugrundeliegende Modell.
In unserem Schokoladenbeispiel beträgt der p-Wert bei einer Devianz von 2.438 und 3 Freiheitsgraden rund 0.49 (49%) und ist somit eindeutig zu groß um am Modell zweifeln zu müssen. Wenn wir in Abbildung 1 den Wert der Devianz auf der x-Achse suchen und von diesem Punkt eine senkrechte Linie nach oben ziehen, dann erhalten wir einen Flächenanteil unter der Verteilungskurve, rechts von der farbigen Linie, welcher 49% beträgt. Das bedeutet, dass (bei Geltung der Nullhypothese) die Wahrscheinlichkeit diesen oder einen größeren Devianzwert zu beobachten, $p = 0.49$ (49%) beträgt – das LLBT-Modell also plausibel ist.

Ad Abschnitt 5.2.1, Seite 33
Datenaufbereitung

1. (Roh)Daten der „Schokoladenverkostung"

Die Datei `schoko.RData` (siehe Kapitel 5.2.1, Seite 31) kann in R mit dem Befehl

```
> load("schoko.RData")
```

eingelesen[1] werden. Die Daten sind dann unter dem Objektnamen `schoko` als Daten Frame verfügbar.

Mit der Funktion `dim()` erhalten wir einen ersten Überblick der Dimension, also der Größe des Daten Frames `schoko`.

[1] siehe dazu auch Seite 33 und Anhang A.1., Seite 199.

```
> dim(schoko)
[1] 100   6
```

Der erste Wert gibt die Anzahl der Zeilen, der zweite die Anzahl der Spalten wieder. Der Daten Frame `schoko` enthält 100 Zeilen (es wurden 100 Personen befragt) und 6 Spalten (eine Spalte pro Paarvergleich).
Mit dem Befehl `head()` können die ersten sechs Zeilen des Daten Frames angezeigt werden.

```
> head(schoko)
  v1 v2 v3 v4 v5 v6
1  1  1  2  1  2  1
2  2  1  2  1  2  2
3  1  1  2  1  1  2
4  1  1  2  2  1  1
5  2  1  2  2  1  2
6  1  1  2  1  1  1
```

Die Zeilen sind Individualdaten, d.h. sie entsprechen den Antwortvektoren der Befragten für sechs Paarvergleiche, von Vergleich 1 (`v1`) bis Vergleich 6 (`v6`), welche die Spalten des Daten Frames darstellen.
Welche Objekte bzw. Gewürzsorten im jeweiligen Paarvergleich miteinander verglichen wurden, ist aus dem Fragebogenausschnitt in Abbildung 5.2, Seite 32 ersichtlich und wird hier nochmals angeführt:
`v1` = Ingwer, Lavendel; `v2` = Minze, Chili; `v3` = Chili, Ingwer; `v4` = Minze, Lavendel; `v5` = Lavendel, Chili und `v6` = Minze, Ingwer.

Die Kodierung der Antworten wurde mit 1 und 2 vorgenommen.
1 das erste Objekt (Gewürz) wird im jeweiligen Paarvergleich bevorzugt,
2 das zweite Objekt (Gewürz) wird im jeweiligen Paarvergleich bevorzugt.

2. Umkodierung der Daten der „Schokoladenverkostung"

Es wird nun gezeigt, wie die Daten der Schokoladenverkostung umkodiert[2] werden können, sodass:

1 für die Bevorzugung des ersten Objekts steht,
– 1 für die Bevorzugung des zweiten Objekts steht.

Der alte Wert 2 soll also durch einen neuen Wert –1 ersetzt werden. Der Wert 1 wird beibehalten:

alter Wert 2 → neuer Wert –1
alter Wert 1 → neuer Wert 1

Die Umkodierung erfolgt mit der Funktion `ifelse()`, wobei die „neuen" Werte direkt in den „alten" Variablen gespeichert werden.
Zunächst wird der Daten Frame `schoko` in eine Matrix mit der Funktion `as.matrix()` umgewandelt und unter `schokom` gespeichert.

```
> schokom <- as.matrix(schoko)
```

i *Die Transformation des Daten Frames in eine Matrix ist notwendig, da die Funktion* `ifelse()` *bei Daten Frames nur für einzelne Variablen funktioniert. Wenn man* `ifelse()` *auf alle Variablen anwenden möchte, dann geht das nur für eine Matrix.*

Dann wird die eigentliche Umkodierung mit der Funktion `ifelse()` durchgeführt und unter `schokoM` gespeichert.

```
> schokoM <- ifelse(schokom == 2, -1, schokom)
```

Bei Ausführung dieser `ifelse`-Funktion passiert Folgendes: für jeden Wert des Daten Frames `schokom` wird überprüft, ob dieser dem Wert 2 entspricht (`==2`). Ist dies der Fall (also `TRUE`), dann wird dieser durch den Wert –1 ersetzt, sonst (also `FALSE`) wird der Wert der im Daten Frame `schokom` enthalten ist beibehalten. Kurz gesagt: Die Funktionswerte der Funktion `ifelse(wenn, dann, sonst)` sind der Reihe nach: wenn, dann, sonst.
Mit der Funktion `head()` kann das Resultat der Umkodierung anhand der ersten sechs Zeilen der Matrix `schokoM` überprüft werden (vgl. Seite 223).

[2] Ziel des Umkodierens in diesem Beispiel ist es, die Werte aller Variablen zu ändern und das Ergebnis in die selben („alten") Variablen zu speichern.

```
> head(schokoM)
  v1 v2 v3 v4 v5 v6
1  1  1 -1  1 -1  1
2 -1  1 -1  1 -1 -1
3  1  1 -1  1  1 -1
4  1  1 -1 -1  1  1
5 -1  1 -1 -1  1 -1
6  1  1 -1  1  1  1
```

3. Transformation der Daten in die „Standardanordnung"

Wie Ihnen vielleicht bereits aufgefallen ist, entsprechen die bei der Gewürzschokoladenverkostung durchgeführten Vergleiche (siehe Fragebogenausschnitt, Seite 32) nicht der Standardanordnung für Paarvergleiche (siehe Kapitel 4, Seite 17).
Die Reihenfolge der Paarvergleiche muss jedoch der Standardreihenfolge entsprechen, um von dem R-Paket prefmod korrekt weiterverarbeitet werden zu können. Daher müssen die Daten zunächst noch transformiert werden. Es muss sowohl die Reihenfolge der Paarvergleiche als auch die Anordnung der Objekte innerhalb der Paarvergleiche der Standardreihenfolge entsprechen. Die Standardreihenfolge der Gewürzschokoladenpaare ist in Tabelle 1 dargestellt.

Tabelle 1.: Standardreihenfolge – Gewürzschokoladenpaare (I=Ingwer, M=Minze, L=Lavendel, C=Chili)

V1	V2	V3	V4	V5	V6
(12)	(13)	(23)	(14)	(24)	(34)
I, M	I, L	M, L	I, C	M, C	L, C

Die Daten werden daher in zwei Schritten transformiert:

1. Änderung der Objektanordnung innerhalb der Vergleiche

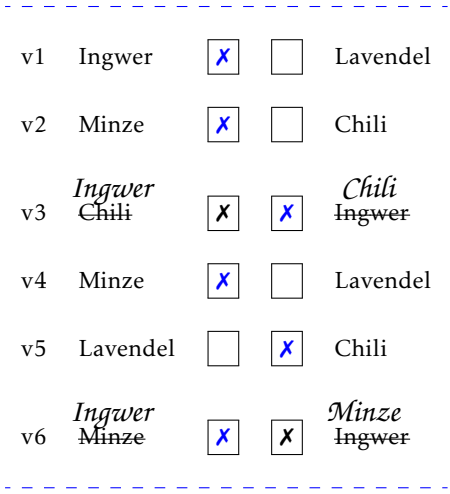

Abbildung 2.: Fragebogenausschnitt – Antworten der ersten befragten Person

Im Vergleich v3 (siehe Abb. 2) muss gemäß der standardmäßigen Objektanordnung innerhalb eines Vergleichs, Ingwer das erste und Chili das zweite Vergleichsobjekt sein und im Vergleich v6 muss das erste Objekt Ingwer lauten und das zweite Objekt Minze.
Demnach müssen alle Werte der Spalten 3 und 6 (die Spalten beziehen sich auf den jeweiligen Paarvergleich) der Matrix `schokoM` mit -1 multipliziert werden (also umgekehrt) werden, damit die Anordnung innerhalb der Paarvergleiche der Standardreihenfolge entspricht. Diese Transformation kann anhand folgender Befehle durchgeführt werden:

```
> schokoM[, 3] <- schokoM[, 3] * -1
> schokoM[, 6] <- schokoM[, 6] * -1
```

Der erste Wert in der eckigen Klammer bezieht sich auf die Zeilen, der zweite Wert auf die Spalten. Diese beiden Werte der eckigen Klammern – getrennt durch ein Komma – werden als Indizes der Matrix (welche aus einem Zeilenindex und einem Spaltenindex besteht) bezeichnet. Da die Transformation über alle Zeilen der Matrix `schokoM` erfolgen soll, wird der Zeilenindex ausgelassen (bzw. nicht eingeschränkt) und nur der interessierende Spaltenindex spezifiziert. In diesem Fall die Spalte 3 und die Spalte 6.

Das Ergebnis dieser Transformation kann z.B. wieder anhand des Befehls `head(schokoM)` für einige Zeilen der Matrix ausgegeben werden.

2. Änderung der Reihenfolge der Paarvergleiche

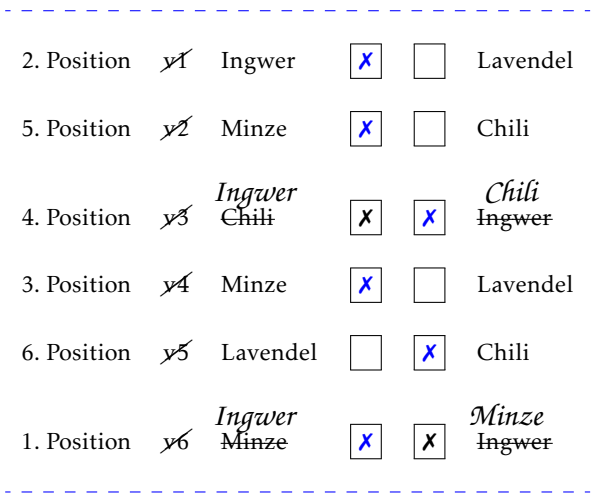

Abbildung 3.: Fragebogenausschnitt – Antworten der ersten befragten Person

Wie in Abbildung 3 im Vergleich zur Tabelle 1 ersichtlich, muss auch die Reihenfolge der Spalten von `v1 bis v6` (Paarvergleiche) der bestehenden Matrix `schokoM` geändert werden, um der Standardreihenfolge zu entsprechen. Der bisherige Vergleich 1 (v1) Ingwer, Lavendel entspricht nicht dem Vergleich 1 laut Standardanordnung. Gemäß der Standardreihenfolge bezieht sich der 1. Vergleich auf den Vergleich zwischen Ingwer und Minze, dem bisherigen Vergleich 6 (v6). Daher müsste Spalte `v6` der Matrix `schokoM` die erste Position bzw. Spalte der Matrix `schokoM1` einnehmen, usw. Die Änderung der Reihenfolge aller Paarvergleiche erfolgt durch:

```
> schokoM1 <- schokoM[, c(6, 1, 4, 3, 2, 5)]
```

Der Spaltenindex wird daher hinsichtlich der Standardreihenfolge als Vektor spezifiziert. Der Vektor `c(6,1,4,3,2,5)` bedeutet, dass die 6. Spalte der Matrix `schokoM` (`v6`) zur **1.** Spalte der „neuen" Matrix **`schokoM1`** wird, die 1. Spalte der Matrix `schokoM` (`v1`) zur **2.** Spalte der „neuen" Matrix **`schokoM1`**, die

4. Spalte der Matrix schokoM (v4) zur **3.** Spalte der Matrix **schokoM1**, ... usw. (siehe Abb. 3 und Tab. 2).

Tabelle 2.: Spezifizierung der Reihenfolge der Spalten für die „neue" Matrix schokoM1

schokoM	v1	v2	v3	v4	v5	v6
sollte sein:	2. Spalte	5. Spalte	4. Spalte	3. Spalte	6. Spalte	1. Spalte
daher:	v6 auf	v1 auf	v4 auf	v3 auf	v2. auf	v5 auf
Sollposition:	Pos. **1**	Pos. **2**	Pos. **3**	Pos. **4**	Pos. **5**	Pos. **6**
	↓	↓	↓	↓	↓	↓
schokoM1	v6	v1	v4	v3	v2	v5

```
> head(schokoM1)
```

```
  v6 v1 v4 v3 v2 v5
1 -1  1  1  1  1 -1
2  1 -1  1  1  1 -1
3  1  1  1  1  1  1
4 -1  1 -1  1  1  1
5  1 -1 -1  1  1  1
6 -1  1  1  1  1  1
```

Nachdem die Spalten der Matrix neu geordnet und unter schokoM1 gespeichert wurden, werden noch die Spaltennamen mit der Funktion colnames() geändert:

```
> colnames(schokoM1) <- c("V1", "V2", "V3",
+     "V4", "V5", "V6")
```

4. Erzeugung eines Daten Frames

In einem letzten Schritt wird die vorliegende Matrix schokoM1 wieder in ein Daten Frame mit dem Befehl

```
> schoko1 <- data.frame(schokoM1)
```

umgewandelt und der Daten Frame unter dem Objektnamen `schoko1` gespeichert.
Das Endresultat der Datenvorbereitungen kann wieder mit dem Befehl `head()` für die ersten sechs Zeilen dargestellt werden (vgl. auch Seite 33):

```
> head(schoko1)
  V1 V2 V3 V4 V5 V6
1 -1  1  1  1  1 -1
2  1 -1  1  1  1 -1
3  1  1  1  1  1  1
4 -1  1 -1  1  1  1
5  1 -1 -1  1  1  1
6 -1  1  1  1  1  1
```

Ad Abschnitt 5.2.3, Seite 36 und 38
Modellschätzung für das Basismodell

Modellschätzung *Möglichkeit 2*: `llbt.design()`, `gnm()`

Der erste Schritt ist die Erstellung einer Designstruktur, welche benötigt wird, um später das gewünschte LLBT Modell mittels der Funktion `gnm()` berechnen zu können.
Zur Erstellung der Designstruktur verwenden wir die Funktion `llbt.design()`.

Designstruktur

▹ **Funktion:**

```
llbt.design(data, nitems = ..., objnames = "")
```

Als erster Funktionswert (`data`) wird ein Daten Frame oder der Name eines Datenfiles angegeben. Die Daten bzw. Antworten müssen so kodiert sein, dass niedrigere Werte für die Präferenz des ersten Objekts in einem Paarvergleich stehen (z.B. {1,2} wobei 1 bedeutet, dass das erste Objekt im Vergleich bevorzugt wurde). Nur im Falle der Kodierung {1,−1} steht 1 für die Präferenz des ersten Objekts und −1 für die Präferenz des zweiten Objekts im jeweiligen Paarvergleich. Der zweite Funktionswert, nämlich die Anzahl der Items bzw. Objekte (`nitems`), muss auf jeden Fall spezifiziert werden und die Option `objnames` bezieht sich auf die Objektnamen, welche, wenn sie nicht

spezifiziert werden, mit `o1, o2, ...` usw. je nach Anzahl der Objekte benannt werden.

! *Bei der Funktion* `llbtPC.fit()` *werden die Objektnamen mit* `obj.names` *festgelegt, bei der Funktion* `llbt.design()` *wird* `objnames` *hingegen ohne Punkt dazwischen geschrieben.*

Als Funktionswerte werden die Datendatei `schoko1` und die Anzahl der Objekte `nitems=4` übergeben. Die Designstruktur wird mit dem Befehl

```
> des <- llbt.design(schoko1, nitems = 4)
```

erstellt und im Objekt `des` gespeichert.

```
> des
    y mu g0 g1 o1 o2 o3 o4
1  50  1  1  0  1 -1  0  0
2  50  1  0  1 -1  1  0  0
3  67  2  1  0  1  0 -1  0
4  33  2  0  1 -1  0  1  0
5  58  3  1  0  0  1 -1  0
6  42  3  0  1  0 -1  1  0
7  91  4  1  0  1  0  0 -1
8   9  4  0  1 -1  0  0  1
9  91  5  1  0  0  1  0 -1
10  9  5  0  1  0 -1  0  1
11 80  6  1  0  0  0  1 -1
12 20  6  0  1  0  0 -1  1
```

Als Resultat erhält man ein Daten Frame für ein LLBT Modell mit der Anzahl der Bevorzugungen (hier bezeichnet mit `y`) in der ersten Spalte und der Designmatrix (siehe auch Designmatrix auf Seite 23).

Die Designstruktur des besteht aus folgenden Spalten:

- `y`: Anzahl der Bevorzugungen für bestimmte Objekte in bestimmten Paarvergleichen. Sehen wir uns beispielsweise den zweiten Vergleich (`mu=2`), also den Vergleich zwischen Objekt 1 (Ingwer) und Objekt 3 (Lavendel), näher an (3. und 4. Zeile der Designstruktur `des`). Die 3. Zeile bezieht sich auf die Bevorzugung von Ingwer und die 4. Zeile auf die Bevorzugung von Lavendel. In der ersten Spalte `y` kann man beispielsweise ablesen, dass Ingwer (`o1`) von 67 Personen und Lavendel (`o3`) von 33 Personen bevorzugt wird.
- `mu`: Faktor mit einer Stufe pro Paarvergleich (es gibt 6 Paarvergleiche und jeder Paarvergleich wird anhand von zwei Zeilen dargestellt).
- `g0,g1`: Indikatorvariablen für zwei Antwortkategorien. Es könnten Effekte für Kategoriebevorzugungen („Responsesets") modelliert werden, welche bei nur zwei Antwortkategorien unbedeutend sind. Von Interesse ist vor allem die Variable der Kategorie „unentschieden", welche Gegenstand des nächsten Anwendungsbeispiels (siehe Abschnitt 6.2) sein wird.
- `o1-o4`: Variablen für die Objekte mit den Ausprägungen 1 für *bevorzugt* und -1 für *nicht bevorzugt* in dem jeweiligen Paarvergleich. 0 bedeutet bei nur zwei Antwortkategorien (*bevorzugt*, *nicht bevorzugt*), dass ein bestimmtes Objekt (z.B. o3) nicht Teil eines bestimmten Paarvergleichs (z.B. o1, o2) ist und somit keinen Eintrag erhält.

Möchte man die Objektnamen ändern, so kann dies direkt bei Erstellung der Designstruktur mit Spezifikation des Befehls `objnames` erfolgen. Anstatt o1, o2 usw. sollen die Objekte (Schokoladen) nach den Anfangsbuchstaben der Gewürzsorte benannt werden, also Ingwer (I), Minze (M), Lavendel (L) und Chili (C).

```
> des2 <- llbt.design(schoko1, nitems = 4, objnames = c("I",
+     "M", "L", "C"))
```

Modellschätzung

▹ **Funktion:**

```
gnm(formula, eliminate = ..., family = ..., data = ...)
```

Um nun auf Basis der vorliegenden Designstruktur `des2` ein LLBT Modell berechnen zu können, wird die Funktion `gnm()` verwendet und das Resultat im Objekt `res2` gespeichert.

```
> res2 <- gnm(y ~ I + M + L + C, eliminate = mu,
+     family = poisson, data = des2)
```

Die Modellformel (`formula`) wird mit `y ~ I + M + L + C` spezifiziert. Die Spezifikation der Option `eliminate` mit `eliminate=mu` bedeutet, dass die nicht interessierenden Nuisance-Parameter μ berechnet werden, aber im Output der Modellschätzung nicht erscheinen. Als nächstes wird die Verteilungsfunktion des Modells festgelegt (siehe auch Abschnitt 5.1.2). Per Voreinstellung ist die Option `family` mit `family=gaussian` festgelegt. Da im LLBT Modell eine bedingte Poisson-Verteilung angenommen wird, lautet die Option `family=poisson`. Zum Schluss wird mit `data = des2` noch der Daten Frame `des2` als Funktionswert übergeben. Das Resultat wird im Objekt `res2` gespeichert.

Ausgabe der Modellschätzung

Das Resultat der Modellschätzung kann, wie bei Möglichkeit 1 (Berechnung durch `llbtPC.fit()`), entweder mittels

```
> res2
Call:

gnm(formula = y ~ I + M + L + C, eliminate = mu, family = poisson,
    data = des2)

Coefficients of interest:
     I       M       L       C
1.1025  1.0479  0.7886      NA

Deviance:            2.438264
Pearson chi-squared: 2.450302
Residual df:         3
```

oder mit `summary()`, die mehr Informationen bietet, ausgegeben werden:

```
> summary(res2)
Call:

gnm(formula = y ~ I + M + L + C, eliminate = mu, family = poisson,
    data = des2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6085  -0.3335  -0.1010   0.2647   0.7510

Coefficients of interest:
  Estimate Std. Error z value Pr(>|z|)
I  1.10252    0.10042  10.979   <2e-16 ***
M  1.04788    0.09942  10.540   <2e-16 ***
L  0.78860    0.09537   8.269   <2e-16 ***
C  0.00000         NA      NA       NA
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 2.4383 on 3 degrees of freedom
AIC: 86.682

Number of iterations: 4
```

Wenn man das Resultat des berechneten Modells mit jenem auf Seite 38 vergleicht, erkennt man, dass beide Möglichkeiten der Berechnung eines LLBT Modells zu den selben Ergebnissen führen.
Anhand des Outputs können wieder die berechneten Objektparameter (also die $\hat{\lambda}$'s) der einzelnen Gewürzsorten (I, M, L, C) abgelesen werden.
Wie im R-Output ersichtlich, wurde die Gewürzsorte C (Chili), welche in der Modellformel `y ~ I + M + L + C` als letztes Objekt spezifiziert wurde, standardmäßig als Referenzobjekt herangezogen und somit auf 0 (`NA`) gesetzt. Es wurde also kein Parameter für die Gewürzsorte Chili (C) und somit auch kein Standardfehler berechnet.

Wie kann ein anderes Objekt als Referenzobjekt definiert werden?

Um ein anderes Objekt als Referenzobjekt zu definieren, wird einfach die Reihenfolge der Modellformel geändert, indem das gewünschte Objekt an die

letzte Stelle gesetzt wird.
Möchte man beispielsweise das Objekt L (Lavendel) als Referenzobjekt heranziehen, so lautet der Befehl dafür:

```
> res2L <- gnm(y ~ I + M + C + L, eliminate = mu,
+     family = poisson, data = des2)
> res2L
Call:

gnm(formula = y ~ I + M + C + L, eliminate = mu, family = poisson,
    data = des2)

Coefficients of interest:
      I       M       C       L
 0.3139  0.2593 -0.7886      NA

Deviance:            2.438264
Pearson chi-squared: 2.450302
Residual df:         3
```

Alternativ führt auch die Spezifikation der Option `constrain` mit `"L"` zu dem Ergebnis, dass das Objekt L (Lavendel) als Referenzobjekt herangezogen wird:

```
> res2cL <- gnm(y ~ I + M + L + C, eliminate = mu,
+     family = poisson, data = des2, constrain = "L")
> res2cL
Call:

gnm(formula = y ~ I + M + L + C, eliminate = mu, constrain = "L",
    family = poisson, data = des2)

Coefficients of interest:
      I       M       L       C
 0.3139  0.2593      NA -0.7886

Deviance:            2.438264
Pearson chi-squared: 2.450302
Residual df:         3
```

Wie in diesem Output ersichtlich, wurde für das Objekt L (Lavendel) kein Parameter (`NA`) geschätzt und dieses somit als Referenzobjekt herangezogen. Das Referenzobjekt steht hier jedoch nicht an letzter Stelle, sondern es wird

in diesem Fall die in der Modellformel spezifizierte Objektreihenfolge eingehalten.

Visualisierung der berechneten Objekt- und Werteparameter

Die berechneten Objekt- und Werteparameter können mit der Funktion `llbt.worth()` ausgegeben werden (siehe dazu auch Abschnitt 5.2.4, Seite 39 ff). Für die Objektparameter lautet der Befehl

```
> lambda2 <- llbt.worth(res2, outmat = "lambda")
> lambda2
```

```
   estimate
I 1.1025249
M 1.0478758
L 0.7886026
C 0.0000000
attr(,"class")
[1] "wmat"   "matrix"
```

und für die Werteparameter

```
> worth2 <- llbt.worth(res2)
> worth2
```

```
       worth
I 0.39363118
M 0.35287582
L 0.21009711
C 0.04339588
attr(,"class")
[1] "wmat"   "matrix"
```

So wie bei der Berechnungsmöglichkeit 1 (`llbtPC.fit()`), können die geschätzten Objekt- und Werteparameter mit der Funktion `plot()` dargestellt werden. Der Befehl zur Erstellung einer Präferenzskala mit den geschätzten Objektparametern lautet `plot(lambda2)`. Für die Werteparameter lautet der Befehl `plot(worth2)`. Erweiterungsmöglichkeiten der `plot()`-Funktion sind in Abschnitt 5.2.4, Seite 42 erläutert und gelten auch für die geschätzten Parameter des Modells `res2`, die aus der 2. Berechnungsmöglichkeit mittels `llbt.design()` und `gnm()` resultieren.

Ad Abschnitt 6.2.3, Seite 56
Modellschätzung für das Basismodell mit *ties*

Modellschätzung *Möglichkeit 2*: `llbt.design(), gnm()`

Zunächst wird wieder eine Designstruktur generiert, um dann ein LLBT Modell mit *ties* (unentschieden) mit Hilfe der Funktion `gnm()` berechnen zu können.

Der Befehl zur Erstellung der Designstruktur lautet:

```
> des2 <- llbt.design(cemspc, nitems = 6, objnames = c("LO",
+     "PA", "MI", "SG", "BA", "ST"))
```

Mit der Funktion `head(des2,10)` können die ersten zehn Zeilen der Designstruktur ausgegeben werden.

```
> head(des2, 10)
     y mu g0 g1 g2 LO PA MI SG BA ST
1  186  1  1  0  0  1 -1  0  0  0  0
2   26  1  0  1  0  0  0  0  0  0  0
3   91  1  0  0  1 -1  1  0  0  0  0
4  221  2  1  0  0  1  0 -1  0  0  0
5   26  2  0  1  0  0  0  0  0  0  0
6   56  2  0  0  1 -1  0  1  0  0  0
7  121  3  1  0  0  0  1 -1  0  0  0
8   32  3  0  1  0  0  0  0  0  0  0
9   59  3  0  0  1  0 -1  1  0  0  0
10 208  4  1  0  0  1  0  0 -1  0  0
```

Die Funktion `llbt.design()` erkennt automatisch, dass es sich um einen Datensatz mit drei Antwortkategorien handelt und berücksichtigt alle Antwortkategorien (bezeichnet mit `g0, g1, g2`) bei Generierung der Designstruktur. Die Variable `g0` ist eine Indikatorvariable für die Bevorzugung des ersten Objekts im jeweiligen Paarvergleich, `g1` für die Kategorie *„unentschieden"* und `g2` für die Bevorzugung des zweiten Objekts im jeweiligen Paarvergleich. Sehen wir uns beispielsweise die 2. Zeile der Designstruktur `des2` an. Die zweite Zeile bezieht sich auf die Antwortkategorie „unentschieden" (`g1=1`) im ersten Paarvergleich (`mu=1`) zwischen London (`LO`) und Paris (`PA`). Da keine der beiden Städte bevorzugt wird (26 Befragte also keine Präferenz für eine

der beiden Städte haben), enthalten die Spalten LO und PA jeweils den Eintrag 0.

Modellschätzung

Um die interessierenden Objektparameter $\lambda_{LO}, \lambda_{PA}, \lambda_{MI}, \lambda_{SG}, \lambda_{BA}, \lambda_{ST}$ sowie den Parameter γ des LLBT Modells für *ties* berechnen zu können, wird die Funktion gnm() verwendet. Die Modellformel (formula) des LLBT Modells mit Berücksichtigung des Effektes der Antwortkategorie „unentschieden", wird festgelegt mit y ~ LO + PA + MI + SG + BA + ST + g1. Im Gegensatz zum LLBT Basismodell wird die Modellformel um g1 erweitert. Der eliminate-Term wird wieder mit mu spezifiziert. Als nächstes wird die Verteilung mit family=poisson spezifiziert und schließlich wird dem Argument (data) die Designstruktur, welche unter dem Objektnamen des2 gespeichert ist, übergeben. Das Ergebnis der Modellberechnung wird unter dem Namen res2 gespeichert.

```
> res2 <- gnm(y ~ LO + PA + MI + SG + BA + ST +
+     g1, eliminate = mu, family = poisson,
+     data = des2)
> res2
Call:

gnm(formula = y ~ LO + PA + MI + SG + BA + ST + g1, eliminate = mu,
    family = poisson, data = des2)

Coefficients of interest:
      LO        PA        MI        SG        BA
 0.79062   0.39743   0.10450   0.18196   0.08047
      ST        g1
      NA  -1.32619

Deviance:            140.4829
Pearson chi-squared: 142.7013
Residual df:         24
```

Ausgabe, Visualisierung der berechneten Objekt- und Werteparameter

```
> lambda2 <- llbt.worth(res2, outmat = "lambda")
```

Auch mit Hilfe des Befehls parameters(res2) können beispielsweise die geschätzten Objektparameter des Modells res2 ausgegeben werden.

```
> parameters(res2)
Coefficients of interest:
         LO          PA          MI          SG
 0.79062281  0.39742862  0.10450057  0.18196020
         BA          ST          g1
 0.08046558  0.00000000 -1.32619038
```

Die Einträge 1 bis 6 beziehen sich auf die geschätzten Objektparameter und der 7. Eintrag (g1) auf den geschätzten γ-Parameter. Möchte man auf den $\hat{\gamma}$-Parameter, der die Antwortkategorie „unentschieden“ repräsentiert, zugreifen so kann dies mit folgendem Befehl erfolgen:

```
> gamma <- parameters(res2)[7]
> gamma
      g1
-1.32619
```

Für die Ausgabe der berechneten Werteparameter verwenden wir den Befehl:

```
> worth2 <- llbt.worth(res2)
```

Die Visualisierung der geschätzten Parameter kann einfach mit der Funktion `plot()` erfolgen. Für die geschätzten Objektparameter mittels `plot(lambda2)` und für die geschätzten Werteparameter mittels `plot(worth2)`.

Ad Abschnitt 7.2.3, Seite 71
Modellschätzung mit *ties* und einer Subjektkovariate

Modellschätzung *Möglichkeit 2*: `llbt.design()`, `gnm()`

Um ein LLBT Modell mit *ties* und einer Subjektkovariate mit Hilfe der Funktion `gnm()` berechnen zu können und somit die interessierenden Parameter zu erhalten, muss zunächst wieder eine Designstruktur mit der Funktion `llbt.design()` erstellt werden.

Designstruktur

▹ **Funktion:**

```
llbt.design(data, nitems = ..., objnames = "",
   cat.scovs = ...)
```

Als erstes Argument (data) werden die Daten cemspc übergeben, dann muss die Anzahl der Objekte (nitems) spezifiziert werden. Anhand der Option objnames können Namen für die jeweiligen Objekte vergeben werden. „Neu" für die Berücksichtigung von Subjektkovariaten in der Designstruktur ist die Option cat.scovs, welche ebenso spezifiziert werden muss. Die Bezeichnung cat.scovs steht für kategoriale Subjektkovariaten. Anhand dieser Option werden jene Subjektkovariaten festgelegt, die in der Designstruktur enthalten sein sollen. Zunächst interessiert uns in diesem Beispiel nur die Subjektkovariate Geschlecht (SEX) und daher übergeben wir als Funktionswert cat.scov = c("SEX").

```
> des2 <- llbt.design(cemspc, nitems = 6, objnames = c("LO",
+     "PA", "MI", "SG", "BA", "ST"), cat.scov = c("SEX"))
> head(des2)
    y mu g0 g1 g2 LO PA MI SG BA ST SEX
1  91  1  1  0  0  1 -1  0  0  0  0   1
2  10  1  0  1  0  0  0  0  0  0  0   1
3  51  1  0  0  1 -1  1  0  0  0  0   1
4 102  2  1  0  0  1  0 -1  0  0  0   1
5  12  2  0  1  0  0  0  0  0  0  0   1
6  38  2  0  0  1 -1  0  1  0  0  0   1
```

Wie anhand der ersten sechs Zeilen des Outputs der Designstruktur des2 ersichtlich, wurde die Subjektkovariate SEX als letzte Spalte des Daten Frames des2 ausgewiesen und mit dem ersten Level (SEX=1) beginnend gereiht. Die Designstruktur für jeden Level dieser Subjektkovariate besteht aus $\binom{J}{2} \cdot 3 = 15 \cdot 3 = 45$ Zeilen (Anzahl der Paarvergleiche x Anzahl der Antwortkategorien). Der erste Level (weibliche Studierende) bezieht sich auf die Zeilen 1-45. Der zweite Level (männliche Studierende, SEX=2) umfasst die Zeilen 46-90. Insgesamt besteht die Designstruktur des2 aus 90 Zeilen. Mit der Funktion tail() können die letzten sechs Zeilen der Designstruktur des2 ausgegeben werden.

```
> tail(des2)
    y mu g0 g1 g2 LO PA MI SG BA ST SEX
85 75 14  1  0  0  0  0  0  1  0 -1   2
86 28 14  0  1  0  0  0  0  0  0  0   2
87 48 14  0  0  1  0  0  0 -1  0  1   2
88 49 15  1  0  0  0  0  0  0  1 -1   2
89 25 15  0  1  0  0  0  0  0  0  0   2
90 77 15  0  0  1  0  0  0  0 -1  1   2
```

Modellschätzung

Auf Basis der erstellten Designstruktur `des2` erfolgt nun die Modellschätzung mit Hilfe der Funktion `gnm()`. Als erstes Argument (`formula`) der Funktion `gnm()` wird die Modellformel spezifiziert. Die Modellformel für das Modell mit *ties* und einer kategorialen Subjektkovariate `SEX` lautet:
y ~ LO+PA+MI+SG+BA+ST+(LO+PA+MI+SG+BA+ST):SEX+ g1 .

i *Der Operator* `:` *steht für eine Interaktion (Wechselwirkung). Die Subjekt-Objekt Wechselwirkungsparameter erhält man durch die Interaktionen* (:) *zwischen den Universitäten und dem Geschlecht. Der Term* (`LO+PA+MI+SG+BA+ST`):`SEX` *ist äquivalent zu der Spezifikation* `LO:SEX + PA:SEX + ... + ST:SEX`.

Zusätzlich zur Erweiterung der Modellformel um Subjekt-Objekt Interaktionsterme ist auch eine, um die Subjektkovariate `SEX`, erweiterte Spezifikation der Option `eliminate` (`eliminate=mu:SEX`) notwendig. Die Spezifikation des `eliminate`-Terms wird ganz allgemein formuliert benötigt, um die Dimension der Designstruktur, für die das Modell berechnet wird, festzulegen.
Die weiteren Argumente der Funktion `gnm()` sollten bereits aus den vorhergehenden Kapiteln bekannt sein und werden daher nicht näher erläutert.
Die Modellschätzung mittels der Funktion `gnm()` erfolgt durch den Befehl:

```
> res2 <- gnm(y ~ LO + PA + MI + SG + BA + ST +
+     (LO + PA + MI + SG + BA + ST):SEX + g1,
+     elim = mu:SEX, family = poisson, data = des2)
> res2
Call:

gnm(formula = y ~ LO + PA + MI + SG + BA + ST + (LO + PA + MI +
    SG + BA + ST):SEX + g1, eliminate = mu:SEX, family = poisson,
    data = des2)

Coefficients of interest:
       LO         PA         MI         SG         BA
 0.844413   0.517704   0.260481   0.150515   0.078805
       ST         g1     LO:SEX2    PA:SEX2    MI:SEX2
       NA  -1.320396  -0.099472  -0.234178  -0.311767
  SG:SEX2    BA:SEX2    ST:SEX2
```

```
 0.064760   0.004294         NA

Deviance:            172.7383
Pearson chi-squared: 179.7135
Residual df:         49
```

Ausgabe und Visualisierung der geschätzten Objekt- und Werteparameter getrennt für alle Level der Subjektkovariate

```
> lambda2 <- llbt.worth(res2, outmat = "lambda")
> lambda2
         SEX1        SEX2
LO 0.84441250  0.74494082
PA 0.51770366  0.28352560
MI 0.26048053 -0.05128620
SG 0.15051504  0.21527539
BA 0.07880496  0.08309931
ST 0.00000000  0.00000000
attr(,"class")
[1] "wmat"   "matrix"
> worth2 <- llbt.worth(res2)
> worth2
         SEX1       SEX2
LO 0.40291232 0.40999423
PA 0.20962103 0.16292869
MI 0.12531807 0.08340340
SG 0.10057704 0.14213996
BA 0.08713894 0.10912129
ST 0.07443260 0.09241243
attr(,"class")
[1] "wmat"   "matrix"


> plot(lambda2)
> plot(worth2)
```

Ad Abschnitt 8.2, Seite 86
Modellschätzung mit zwei Subjektkovariaten

Modellschätzung *Möglichkeit 2*: `llbt.design()`, `gnm()`

```
> des2 <- llbt.design(cemspc, nitems = 6, objnames = c("LO",
+     "PA", "MI", "SG", "BA", "ST"), cat.scov = c("ENG",
+     "SEX"))
> m2es <- gnm(y ~ LO + PA + MI + SG + BA + ST +
+     (LO + PA + MI + SG + BA + ST):(ENG * SEX) +
+     g1, elim = mu:ENG:SEX, data = des2, family = poisson)
```

In diesem Beispiel ist das hierarchisch höchste Modell das Wechselwirkungsmodell mit dem Interaktionsterm `ENG*SEX`, danach folgt das additive Modell (`ENG+SEX`), die beiden Haupteffektmodelle (`ENG`) und (`SEX`), sowie das einfachste Modell, das Nullmodell (`1`).
Die für die Schätzung mittels `gnm()` benötigte Designstruktur wurde bereits unter dem Namen `des2` temporär gespeichert und kann der Funktion `gnm()` einfach übergeben werden.

- Modell für die Wechselwirkung `ENG*SEX`

  Dieses Modell (das hierarchisch höchste) wurde bereits unter dem Objektnamen `m2es` gespeichert.

- Additives Modell für die beiden Haupteffekte `ENG+SEX`

```
> m2engsex <- gnm(y ~ LO + PA + MI + SG + BA +
+     ST + (LO + PA + MI + SG + BA + ST):(ENG +
+     SEX) + g1, elim = mu:ENG:SEX, data = des2,
+     family = poisson)
```

- Haupteffektmodell für `ENG`

```
> m2eng <- gnm(y ~ LO + PA + MI + SG + BA +
+     ST + (LO + PA + MI + SG + BA + ST):ENG +
+     g1, elim = mu:ENG:SEX, data = des2, family = poisson)
```

- Haupteffektmodell für `SEX`

```
> m2sex <- gnm(y ~ LO + PA + MI + SG + BA +
+     ST + (LO + PA + MI + SG + BA + ST):SEX +
+     g1, elim = mu:ENG:SEX, data = des2, family = poisson)
```

- Nullmodell

```
> m2null <- gnm(y ~ LO + PA + MI + SG + BA +
+      ST + g1, elim = mu:ENG:SEX, data = des2,
+      family = poisson)
```

Modellselektion (siehe auch Seite 88)

```
> anova(m2es, m2engsex, m2null, test = "Chisq")
> anova(m2engsex, m2eng, test = "Chisq")
> anova(m2engsex, m2sex, test = "Chisq")
```

Das finale Modell ist das Modell `m2engsex`.

```
> m2engsex
Call:

gnm(formula = y ~ LO + PA + MI + SG + BA + ST + (LO + PA + MI +
    SG + BA + ST):(ENG + SEX) + g1, eliminate = mu:ENG:SEX, family = poisson,
    data = des2)

Coefficients of interest:
       LO         PA         MI         SG         BA
 0.853989   0.548408   0.254985   0.105477   0.097467
       ST         g1    LO:ENG2    LO:SEX2    PA:ENG2
       NA  -1.318135  -0.032819  -0.097949  -0.121578
  PA:SEX2    MI:ENG2    MI:SEX2    SG:ENG2    SG:SEX2
-0.229270   0.026212  -0.313433   0.182960   0.057081
  BA:ENG2    BA:SEX2    ST:ENG2    ST:SEX2
-0.076280   0.007946         NA         NA

Deviance:            219.4753
Pearson chi-squared: 225.3611
Residual df:         104
```

In R können die berechneten Objektparameter des Modells `m2engsex` für die vier Gruppen (`ENG1:SEX1`, `ENG2:SEX1`, `ENG1:SEX2`, `ENG2:SEX2`) einfach mit der Funktion `llbt.worth()` ausgegeben werden.

```
> lambda_m2engsex <- llbt.worth(m2engsex, outmat = "lambda")
> lambda_m2engsex
```

```
    ENG1:SEX1  ENG2:SEX1   ENG1:SEX2   ENG2:SEX2
LO 0.85398947 0.82117080  0.75604092  0.72322225
PA 0.54840758 0.42682913  0.31913756  0.19755911
MI 0.25498466 0.28119621 -0.05844871 -0.03223716
SG 0.10547738 0.28843772  0.16255816  0.34551850
BA 0.09746748 0.02118773  0.10541320  0.02913345
ST 0.00000000 0.00000000  0.00000000  0.00000000
attr(,"class")
[1] "wmat"   "matrix"
```

Die berechneten Werteparameter können wieder anhand folgendem Befehl ausgegeben werden:

```
> worth_m2engsex <- llbt.worth(m2engsex)
> worth_m2engsex
    ENG1:SEX1  ENG2:SEX1  ENG1:SEX2  ENG2:SEX2
LO 0.40489459 0.39462482 0.41471824 0.39604971
PA 0.21974395 0.17933439 0.17308670 0.13840905
MI 0.12219488 0.13401964 0.08133855 0.08741098
SG 0.09061344 0.13597477 0.12654954 0.18607173
BA 0.08917340 0.07967608 0.11288210 0.09882614
ST 0.07337974 0.07637030 0.09142487 0.09323239
attr(,"class")
[1] "wmat"   "matrix"
```

Und die Visualisierung der geschätzten Objekt- und Werteparameter kann einfach mit `plot()` erfolgen:

```
> plot(lambda_m2engsex)
> plot(worth_m2engsex)
```

Ad Abschnitt 8.2.3, Seite 91
Berechnung der Objektparameter für 4 Gruppen

In Tabelle 3 ist die Berechnung der Objektparameter für vier Gruppen dargestellt:

Tabelle 3.: Berechnung der Objektparameter für jede der vier Gruppen

ENG1:SEX1	ENG2:SEX1	ENG1:SEX2	ENG2:SEX2
$\hat{\lambda}_{LO}$	$\hat{\lambda}_{LO} + \hat{\lambda}_{LO:ENG2}$	$\hat{\lambda}_{LO} + \hat{\lambda}_{LO:SEX2}$	$\hat{\lambda}_{LO} + \hat{\lambda}_{LO:ENG2} + \hat{\lambda}_{LO:SEX2}$
$\hat{\lambda}_{PA}$	$\hat{\lambda}_{PA} + \hat{\lambda}_{PA:ENG2}$	$\hat{\lambda}_{PA} + \hat{\lambda}_{PA:SEX2}$	$\hat{\lambda}_{PA} + \hat{\lambda}_{PA:ENG2} + \hat{\lambda}_{PA:SEX2}$
$\hat{\lambda}_{MI}$	$\hat{\lambda}_{MI} + \hat{\lambda}_{MI:ENG2}$	$\hat{\lambda}_{MI} + \hat{\lambda}_{MI:SEX2}$	$\hat{\lambda}_{MI} + \hat{\lambda}_{MI:ENG2} + \hat{\lambda}_{MI:SEX2}$
$\hat{\lambda}_{SG}$	$\hat{\lambda}_{SG} + \hat{\lambda}_{SG:ENG2}$	$\hat{\lambda}_{SG} + \hat{\lambda}_{SG:SEX2}$	$\hat{\lambda}_{SG} + \hat{\lambda}_{SG:ENG2} + \hat{\lambda}_{SG:SEX2}$
$\hat{\lambda}_{BA}$	$\hat{\lambda}_{BA} + \hat{\lambda}_{BA:ENG2}$	$\hat{\lambda}_{BA} + \hat{\lambda}_{BA:SEX2}$	$\hat{\lambda}_{BA} + \hat{\lambda}_{BA:ENG2} + \hat{\lambda}_{BA:SEX2}$
$\hat{\lambda}_{ST}$	$\hat{\lambda}_{ST} + \hat{\lambda}_{ST:ENG2}$	$\hat{\lambda}_{ST} + \hat{\lambda}_{ST:SEX2}$	$\hat{\lambda}_{ST} + \hat{\lambda}_{ST:ENG2} + \hat{\lambda}_{ST:SEX2}$

Wir möchten beispielsweise die Präferenzwerte der Universität in London (LO) für die vier Gruppen (`ENG1:SEX1`, `ENG2:SEX1`, `ENG1:SEX2`, `ENG2:SEX2`) berechnen. Für die Gruppe der Befragten die bei den beiden Subjektkovariaten (`ENG` und `SEX`) dem ersten Level zugeordnet werden können (Gruppe `ENG1:SEX1`, weibliche Studierende mit guten Englischkenntnissen), ist der gesuchte Objektparameter der „Referenzparameter“ $\hat{\lambda}_{LO} = 0.854$ (siehe Output des Modells `m2engsex`). Für die Gruppe der weiblichen Studierenden mit schlechten Englischkenntnissen (`ENG2:SEX1`) berechnet sich der Objektparameter durch Addition des Interaktionsparameters $\hat{\lambda}_{LO:ENG2}$ zum geschätzten Referenzparameter, 0.854-0.033=0.821 (vgl. Output `lambda_m2engsex`, Spalte 2, Zeile 1). Für die Gruppe der männlichen Studierenden mit guten Englischkenntnissen (`ENG1:SEX2`) addieren wir den Interaktionsparameter $\hat{\lambda}_{LO:SEX2}$ zum geschätzten Referenzparameter (0.854-0.098=0.756). Schließlich kann der Wert des Objektparameters für die Universität in London für die Gruppe der männlichen Studierenden mit schlechten Englischkenntnissen (`ENG2:SEX2`) durch Addition der beiden Parameter $\hat{\lambda}_{LO:ENG2}$ und $\hat{\lambda}_{LO:SEX2}$ zu $\hat{\lambda}_{LO}$ berechnet werden (0.854-0.033-0.098=0.723).

Ad Abschnitt 12.1.1, Seite 133
Kovariaten x der Objektparameter im Pattern Modell

BEISPIEL

Nehmen wir beispielsweise das Antwortmuster $\boldsymbol{y}_1 = (111)$ das in der ersten Zeile der Designstruktur in Tabelle 12.1 auf Seite 134 abgebildet ist her. Anhand des Antwortmusters können wir herauslesen, dass jeweils das erste Objekt in den drei Paarvergleichen – (12), (13) und (23) – bevorzugt wurde.

Zusammengefasst heißt das, dass Objekt 1 im ersten und im zweiten Vergleich bevorzugt wurde ($x_1 = 1 + 1 = 2$), Objekt 2 im ersten Vergleich nicht und im dritten Vergleich bevorzugt wurde ($x_2 = -1 + 1 = 0$) und Objekt 3 im zweiten und dritten Vergleich nicht bevorzugt wurde ($x_3 = -1 - 1 = -2$). Für das Antwortmuster (111) ergeben sich somit die Einträge (2 0 – 2) als Kovariaten der Objektparameter λ_1, λ_2 und λ_3.

Die Kovariateneinträge x können auch einfach für jedes der möglichen Antwortmuster berechnet werden. Multipliziert man die Matrix mit allen möglichen Antwortmustern mit der Designmatrix für drei Paarvergleiche (siehe Böckenholt & Dillon, 1997) so erhält man eine Matrix mit den gesuchten Kovariaten x_1, x_2, x_3. Nehmen wir als Beispiel nur das Antwortmuster (1 1 1) her und sehen uns die Berechnung näher an:

$$\begin{array}{ccc} y_{12}, y_{13}, y_{23} & & \boldsymbol{B} \\ \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} & \cdot & \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \end{pmatrix} \end{array} \quad = \quad \begin{array}{c} x_1, x_2, x_3 \\ \begin{pmatrix} 2 & 0 & -2 \end{pmatrix} \end{array}$$

Die Spalten der Designmatrix von Böckenholt & Dillon (1997), bezeichnet mit $\boldsymbol{B}$, beziehen sich jeweils auf ein Objekt und die Zeilen auf die jeweiligen Paarvergleiche (bei drei Objekten sind das 3 Paarvergleiche, also 3 Zeilen). Die Berechnung kann durchgeführt werden, indem die Zeile mit dem Antwortmuster (1 1 1) mit der Spalte der Designmatrix $\boldsymbol{B}$ elementweise multipliziert und die Ergebnisse summiert werden, jeweils für alle Spalten der Designmatrix:
$x_1 = (1 \cdot 1) + (1 \cdot 1) + (1 \cdot 0) = 2$
$x_2 = (1 \cdot -1) + (1 \cdot 0) + (1 \cdot 1) = 0$
$x_3 = (1 \cdot 0) + (1 \cdot -1) + (1 \cdot -1) = -2$.

Ad Abschnitt 12.2.1, Seite 136
Modellschätzung für das Pattern Basismodell

Falls das Paket prefmod und die Daten `schoko1.RData` noch nicht geladen sind, werden diese wieder mit folgenden Befehlen in R geladen

```
> library(prefmod)
> load("schoko1.RData")
```

Modellschätzung *Möglichkeit 2*: `patt.design()`, `gnm()`

▹ Funktion:

```
patt.design(obj, nitems = ..., objnames = "")
```

Die Optionen der Funktion `patt.design()` sind ähnlich der Funktion `llbt.design()`. Als erstes Argument (`obj`) wird wieder ein Daten Frame oder der Name eines Datenfiles übergeben und zusätzlich muss die Anzahl der Objekte bzw. Items (`nitems`) festgelegt werden. Möchte man Namen für die zu schätzenden Objekte vergeben, so kann wieder die Option `objnames` spezifiziert werden.

```
> despatt2 <- patt.design(schoko1, nitems = 4,
+     objnames = c("I", "M", "L", "C"))
```

Mit dem Befehl `head()` können wieder die ersten sechs Zeilen der Designstruktur betrachtet werden:

```
> head(despatt2)
   y I  M  L  C
1 12 3  1 -1 -3
2  4 3  1 -3 -1
3  0 3 -1 -1 -1
4  0 3 -1 -3  1
5  0 1  1 -1 -1
6  0 1  1 -3  1
```

▹ Funktion:

```
gnm(formula, eliminate = ..., family = ..., data = ...)
```

Um nun auf Basis der vorliegenden Designstruktur despatt2 ein Pattern Modell berechnen zu können, wird die Funktion `gnm()` verwendet und das Resultat im Objekt `respatt2` gespeichert. Die Modellformel (`formula`) wird mit `y ~ I + M + L + C` spezifiziert. Die Option `family` wird mit `family=poisson` festgelegt und schließlich wird der Daten Frame `despatt2` als Funktionswert übergeben.

```
> respatt2 <- gnm(y ~ I + M + L + C, family = poisson,
+     data = despatt2)
```

Das Resultat der Modellschätzung eines Pattern Modells, das mit der Funktion `gnm()` berechnet wurde, kann mit der Funktion `summary()` überblicksmäßig zusammengefasst und ausgegeben werden.

```
> summary(respatt2)
Call:

gnm(formula = y ~ I + M + L + C, family = poisson, data = despatt2)


Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8771  -0.8770  -0.4553   0.1368   2.7628

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.90482    0.22740  -3.979 6.92e-05 ***
I            1.10252    0.10042  10.979  < 2e-16 ***
M            1.04788    0.09942  10.540  < 2e-16 ***
L            0.78860    0.09537   8.269  < 2e-16 ***
C            0.00000         NA      NA       NA
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 79.48 on 60 degrees of freedom
AIC: 156.23

Number of iterations: 6
```

i *Der $\hat{\mu}$- Parameter ist im* R *-Output unter der Bezeichnung* `Intercept` *ablesbar.*

Ein Vorteil der Funktion `summary()` ist, dass die Freiheitsgrade (`degrees of freedom`) direkt abgelesen werden können.

Ausgabe, Visualisierung der berechneten Objekt- und Werteparameter

Die berechneten Objekt- und Werteparameter können mit der Funktion `patt.worth()` ausgegeben werden (siehe dazu auch Abschnitt 12.2.2, Seite 140). Für die Objektparameter lautet der Befehl

```
> plambda2 <- patt.worth(respatt2, outmat = "lambda")
> plambda2
   estimate
I 1.1025249
M 1.0478758
L 0.7886026
C 0.0000000
attr(,"class")
[1] "wmat"   "matrix"
```

und für die Werteparameter

```
> pworth2 <- patt.worth(respatt2)
> pworth2
       worth
I 0.39363118
M 0.35287582
L 0.21009711
C 0.04339588
attr(,"class")
[1] "wmat"   "matrix"
```

Die geschätzten Objekt- und Werteparameter können, wie bei der Berechnungsmöglichkeit 1 (`pattPC.fit()`), mit der Funktion `plot()` dargestellt werden. Der Befehl zur Erstellung einer Präferenzskala mit den geschätzten Objektparametern lautet `plot(plambda2)` und für die Werteparameter `plot(pworth2)`.

Ad Abschnitt 14.2.2, Seite 151
Modellschätzung für das Pattern Modell mit einer Subjektkovariate und Abhängigkeiten

Modellschätzung *Möglichkeit 2*: `patt.design()`, `gnm()`
Im Folgenden wird ein Pattern Modell mit Abhängigkeiten und einer kategorialen Subjektkovariate berechnet. Zunächst wird eine Designstruktur mit

der Funktion `patt.design()` generiert und diese dann der Funktion `gnm()` übergeben.

Designstruktur

▸ **Funktion:**

```
patt.design(obj, nitems = ..., objnames = "",
    cat.scovs = ..., ia = TRUE)
```

Als erstes Argument (`obj`) wird wieder das Datenfile `sport` übergeben, danach wird die Anzahl der Objekte (`nitems`) mit 3 spezifiziert und Objektnamen festgelegt. Der Option `cat.scovs` wird die kategoriale Subjektkovariate `SEX` übergeben (`cat.scovs=c("SEX")`). Die Option `ia`, welche für die Abhängigkeitsparameter steht, ist per Voreinstellung mit `ia=FALSE` festgelegt. Da Abhängigkeitsparameter in der Modellschätzung mitberücksichtigt werden sollen, wird die Option `ia` auf `TRUE` gesetzt. Der Befehl lautet

```
> despatt2 <- patt.design(sport, nitems = 3,
+     objnames = c("B", "J", "C"), cat.scovs = c("SEX"),
+     ia = TRUE)
> despatt2
```

	y	B	J	C	I12.13	I12.23	I13.23	SEX
1	0	2	0	-2	1	1	1	1
2	6	2	-2	0	1	-1	-1	1
3	0	0	0	0	-1	1	-1	1
4	19	0	-2	2	-1	-1	1	1
5	5	0	2	-2	-1	-1	1	1
6	0	0	0	0	-1	1	-1	1
7	17	-2	2	0	1	-1	-1	1
8	21	-2	0	2	1	1	1	1
9	15	2	0	-2	1	1	1	2
10	7	2	-2	0	1	-1	-1	2
11	2	0	0	0	-1	1	-1	2
12	12	0	-2	2	-1	-1	1	2
13	16	0	2	-2	-1	-1	1	2
14	2	0	0	0	-1	1	-1	2
15	23	-2	2	0	1	-1	-1	2
16	19	-2	0	2	1	1	1	2

Aus drei Paarvergleichen resultieren 8 mögliche Antwortmuster. Für jede Gruppe der Subjektkovariate (`SEX1` = männlich und `SEX2` = weiblich) gibt es also 8 mögliche Antwortmuster und somit besteht die Designstruktur aus

insgesamt 16 Antwortmustern (bzw. 16 Zeilen). Die 5. bis 7. Spalte der Designstruktur despatt2, beziehen sich auf die drei Abhängigkeitsparameter für die Vergleichspaare (12)(13), (12)(23) und (13)(23)[3]. In der letzten Spalte ist die kategoriale Subjektkovariate SEX ersichtlich.

Modellschätzung

Die Modellschätzung erfolgt mit Hilfe der Funktion gnm().

```
> respatt2 <- gnm(y ~ B + J + C + I12.13 + I12.23 +
+     I13.23 + (B + J + C):SEX, eliminate = SEX,
+     family = poisson, data = despatt2)
> summary(respatt2)
```

```
Call:

gnm(formula = y ~ B + J + C + I12.13 + I12.23 + I13.23 + (B +
    J + C):SEX, eliminate = SEX, family = poisson, data = despatt2)


Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0724  -0.8535   0.1231   0.5658   1.1826

Coefficients of interest:
       Estimate Std. Error z value Pr(>|z|)
B       -0.6121     0.1183  -5.172 2.32e-07 ***
J       -0.3451     0.1044  -3.306 0.000946 ***
C        0.0000         NA      NA       NA
I12.13   0.5893     0.1416   4.162 3.16e-05 ***
I12.23  -0.6497     0.1393  -4.663 3.11e-06 ***
I13.23   0.5853     0.1414   4.140 3.48e-05 ***
B:SEX2   0.4984     0.1395   3.572 0.000354 ***
J:SEX2   0.4460     0.1267   3.520 0.000432 ***
C:SEX2   0.0000         NA      NA       NA
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)
```

[3] Das I im Output steht für die Interaktion zwischen jeweils zwei Paarvergleichen, die in einem Objekt übereinstimmen.

```
Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 12.855 on 7 degrees of freedom
AIC: 84.677

Number of iterations: 5
```

Ausgabe und Visualisierung der geschätzten Objekt- und Werteparameter getrennt für beide Gruppen der Subjektkovariate

Die geschätzten Objektparameter können ausgegeben werden mittels

```
> plambda2 <- patt.worth(respatt2, outmat = "lambda")
```

und die geschätzten Werteparamter mittels

```
> pworth2 <- patt.worth(respatt2)
> pworth2
       SEX1      SEX2
B 0.1637423 0.2637628
J 0.2793049 0.4051456
C 0.5569528 0.3310915
attr(,"class")
[1] "wmat"   "matrix"
```

Die Präferenzskalen für beide Gruppen der Subjektkovariate `SEX` können wieder einfach mit der Funktion `plot()` erstellt werden. Für die geschätzten Objektparameter lautet der Befehl `plot(plambda2)` und für die geschätzten Werteparameter `plot(pworth2)`.

Ad Abschnitt 15.2.2, Seite 163
Modellschätzungen für das Pattern Modell mit verschiedenen unentschieden Parametern

Modellschätzungen *Möglichkeit 2 und 3*: `patt.design()`, `gnm()`

Die Designstruktur kann für beide Berechnungsmöglichkeiten mit der Funktion `patt.design()` erstellt werden.

Designstruktur

▹ Funktion:

```
patt.design(obj, nitems, objnames = ..., ia = TRUE)
```

Als erstes Argument (`obj`) wird die Datendatei `cems` übergeben und die Anzahl der Objekte (`nitems`) festgelegt. Zur Spezifizierung der Objektnamen (`objnames`) wird ein Vektor mit den Abkürzungen der drei Universitäten übergeben. Die Option `ia` wird auf `TRUE` gesetzt, um Kovariaten für die Abhängigkeiten bzw. Wechselwirkungen zwischen jeweils zwei Paarvergleichen zu generieren. Die Funktion `patt.design()` erkennt, ob die Daten in Form von zwei oder drei Antwortkategorien vorliegen, sodass für jeden Paarvergleich standardmäßig eine Kovariate für die Kategorie „unentschieden“ (bezeichnet mit `u`, in diesem Beispiel `u12`, `u13` und `u23`) in der Designstruktur berücksichtigt wird.

```
> despatt <- patt.design(cems, nitems = 3, ia = TRUE,
+     objnames = c("LO", "PA", "MI"))
> head(despatt)
  y LO PA MI u12 u13 u23 I12.13 I12.23 I13.23
1 53  2  0 -2   0   0   0      1      1      1
2 22  2 -1 -1   0   0   1      1      0      0
3 31  2 -2  0   0   0   0      1     -1     -1
4  0  1  0 -1   0   1   0      0      1      0
5  3  1 -1  0   0   1   1      0      0      0
6  4  1 -2  1   0   1   0      0     -1      0
```

Die Anzahl der Pattern kann mit dem Befehl `nrow(despatt)` *ausgegeben werden.*

Modellschätzung *Möglichkeit 2*

Um *einen* allgemeinen Effekt der Antwortkategorie „unentschieden” in einem Pattern Modell berücksichtigen zu können, werden die Kovariaten für die unentschieden-Parameter γ für die jeweiligen Paarvergleiche `u12`, `u13` und `u23` (die Spalten 5, 6, 7) mit dem Befehl `rowSums()` zeilenweise summiert und im Objekt `U` temporär gespeichert.

```
> U <- rowSums(despatt[, 5:7])
> head(U)
[1] 0 1 0 1 2 1
```

Modellschätzung

```
> respatt2 <- gnm(y ~ LO + PA + MI + U + I12.13 +
+     I12.23 + I13.23, family = poisson, data = despatt)
> summary(respatt2)
Call:

gnm(formula = y ~ LO + PA + MI + U + I12.13 + I12.23 + I13.23,
    family = poisson, data = despatt)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.4560  -1.2832  -0.3950   0.5657   3.5514

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.11836    0.16326  12.975  < 2e-16 ***
LO           0.55149    0.07007   7.870  < 2e-16 ***
PA           0.24861    0.06328   3.929 8.54e-05 ***
MI           0.00000         NA      NA       NA
U           -1.07533    0.13688  -7.856  < 2e-16 ***
I12.13       0.77140    0.15500   4.977 6.46e-07 ***
I12.23      -0.79835    0.14927  -5.349 8.87e-08 ***
I13.23       0.84117    0.15426   5.453 4.95e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 62.481 on 20 degrees of freedom
AIC: 141.5

Number of iterations: 6
```

Modellschätzung *Möglichkeit 3*

Möchte man für jeden der drei Paarvergleiche ((12), (13) und (23)) einen Effekt der Antwortkategorie „unentschieden“ bei der Modellschätzung berücksichtigen, so lautet der Befehl dazu:

```
> respatt3 <- gnm(y ~ LO + PA + MI + u12 + u13 +
+     u23 + I12.13 + I12.23 + I13.23, family = poisson,
```

```
+      data = despatt)
> summary(respatt3)
Call:

gnm(formula = y ~ LO + PA + MI + u12 + u13 + u23 + I12.13 + I12.23 +
    I13.23, family = poisson, data = despatt)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.0837  -1.2290  -0.1641   0.6116   3.6628

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.12760    0.16268  13.078  < 2e-16 ***
LO           0.54313    0.06935   7.832  < 2e-16 ***
PA           0.25087    0.06381   3.931 8.44e-05 ***
MI           0.00000         NA      NA       NA
u12         -1.26947    0.23592  -5.381 7.41e-08 ***
u13         -1.32863    0.27097  -4.903 9.42e-07 ***
u23         -0.74612    0.20108  -3.710 0.000207 ***
I12.13       0.74812    0.15194   4.924 8.49e-07 ***
I12.23      -0.80150    0.14988  -5.348 8.91e-08 ***
I13.23       0.85266    0.15587   5.470 4.49e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 58.192 on 18 degrees of freedom
AIC: 141.22

Number of iterations: 6
```

Ausgabe und Visualisierung der geschätzten Parameter

Im Folgenden wird die Ausgabe des berechneten allgemeinen γ-Parameters (unentschieden) gezeigt.

```
> coef(respatt2)
Coefficients:
(Intercept)          LO          PA          MI
  2.1183606   0.5514924   0.2486073          NA
```

```
         U      I12.13      I12.23      I13.23
-1.0753348   0.7713990  -0.7983515   0.8411714
```

Der $\hat{\gamma}$-Parameter befindet sich auf der 5. Position des Outputs der Funktion `coef()` und kann mit folgendem Befehl ausgegeben werden:

```
> U2 <- coef(respatt2)[5]
> U2
       U
-1.075335
```

Die $\hat{\gamma}$-Parameter (im Output bezeichnet mit u_{12}, u_{13}, u_{23}) für jeweils einen bestimmten Paaarvergleich ((12), (13) und (23)) befinden sich im vorliegenden Beispiel im `coef(respatt3)` Output auf den Positionen 5 bis 7

```
> coef(respatt3)
Coefficients:
(Intercept)          LO          PA          MI
  2.1275965   0.5431256   0.2508725          NA
        u12         u13         u23      I12.13
 -1.2694674  -1.3286333  -0.7461184   0.7481154
     I12.23      I13.23
 -0.8015047   0.8526614
```

und können ausgegeben werden durch

```
> u3 <- coef(respatt3)[5:7]
> u3
       u12        u13        u23
-1.2694674 -1.3286333 -0.7461184
```

Die negativen $\hat{\gamma}$-Parameter weisen darauf hin, dass es in jedem Paarvergleich eine Tendenz gibt, sich für eines der beiden Objekte zu entscheiden.

Für die Ausgabe der geschätzten Objekt- und Werteparameter der Modelle `respatt2` und `respatt3` kann wieder die Funktion `patt.worth()` herangezogen werden.

```
> plambda2 <- patt.worth(respatt2, outmat = "lambda")
> plambda3 <- patt.worth(respatt3, outmat = "lambda")
```

und

```
> pworth2 <- patt.worth(respatt2)
> pworth3 <- patt.worth(respatt3)
```

Die Visualisierung der geschätzten Werteparameter führen wir wieder einfach mit der `plot()`-Funktion durch

```
> plot(pworth2)
```

bzw.

```
> plot(pworth3)
```

Ad Abschnitt 16.2.2, Seite 176
Modellschätzung für Rangdaten

Modellschätzung *Möglichkeit 2*: `patt.design()`, `gnm()`

Designstruktur
Die Designstruktur für ein Pattern Modell, das auf Rangdaten basiert, kann mittels der Funktion `patt.design()` erstellt werden. Mit Hilfe dieser Funktion wird die Transformation von Rangdaten in Paarvergleichsdaten automatisch durchgeführt.

▹ **Funktion:**

```
patt.design(obj, nitems = ..., objnames = "",
   resptype = "ranking")
```

Als erstes Argument (`obj`) der Funktion `patt.design()` wird die Datendatei `rest` übergeben und die Anzahl der Objekte bzw. Items (`nitems`) wird mit vier spezifiziert. „Neu" ist die Spezifikation der Option `resptype`, die, wenn sie nicht spezifiziert wird, standardmäßig auf `paircomp` (also Paarvergleichsdaten) eingestellt ist. Anhand der Option `resptype` wird nun das Antwortformat mit `resptype="ranking"` festgelegt, da es sich bei den Daten um Rangdaten handelt.

```
> despatt2 <- patt.design(rest, nitems = 4,
+     resptype = "ranking")
> head(despatt2)
```

```
   y Qual Freund Preis Schnell
1 13    3      1    -1      -3
2  7    1      3    -1      -3
3  3    1     -1     3      -3
4 12    3     -1     1      -3
5  3   -1      3     1      -3
6  3   -1      1     3      -3
```

Als Ergebnis resultiert ein Daten Frame, in dem die Häufigkeiten mit denen ein bestimmtes hergeleitetes Antwortmuster auftritt, in der ersten Spalte y dargestellt sind. In den nächsten vier Spalten `Qual`, `Freund`, `Preis` und `Schnell` sind die Werte der Kovariaten (x) der interessierenden Objektparameter, der λ's.

Modellschätzung

Nach Erstellung der Designstruktur erfolgt die Modellschätzung mit folgendem Befehl:

```
> respatt2 <- gnm(y ~ Qual + Freund + Preis +
+     Schnell, family = poisson, data = despatt2)
> summary(respatt2)
Call:

gnm(formula = y ~ Qual + Freund + Preis + Schnell, family = poisson,
    data = despatt2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5463  -0.7682  -0.4282   0.2490   1.2722

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.18724    0.28491  -0.657 0.511070
Qual         0.94773    0.11964   7.921  < 2e-16 ***
Freund       0.46707    0.09056   5.158  2.5e-07 ***
Preis        0.30998    0.08685   3.569 0.000358 ***
Schnell      0.00000         NA      NA       NA
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)
```

```
Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 17.832 on 20 degrees of freedom
AIC: 66.037

Number of iterations: 4
```

Ausgabe, Visualisierung der geschätzten Objekt- und Werteparameter

Die Ausgabe der berechneten Parameter kann wieder mit dem Befehl `patt.wort()` erfolgen. Für die geschätzten Objektparameter

```
> plambda2 <- patt.worth(respatt2, outmat = "lambda")
```

und für die geschätzten Werteparameter

```
> pworth2 <- patt.worth(respatt2)
> pworth2
             worth
Qual    0.55189580
Freund  0.21104077
Preis   0.15414102
Schnell 0.08292241
attr(,"class")
[1] "wmat"   "matrix"
```

Mit dem Befehl

```
> plot(pworth2)
```

können die geschätzten Werteparameter visualisiert werden.

Ad Abschnitt 17.2.2, Seite 192
Modellschätzung für Ratingdaten

Modellschätzungen *Möglichkeit 2 und 3*: `patt.design()`, `gnm()`

Die Designstruktur kann für beide Berechnungsmöglichkeiten mit der Funktion `patt.design()` erstellt werden.

Designstruktur

▹ **Funktion:**

```
patt.design(obj, nitems = ..., objnames = "",
   resptype = "rating")
```

Die Designstruktur wird anhand der Funktion `patt.design()` erzeugt. Zunächst wird wieder der Daten Frame `issp2000` übergeben und die Anzahl der Items mit sechs spezifiziert. Die Option `resptype` ist per Voreinstellung mit `paircomp` festgelegt. Da es sich bei Ratingdaten nicht um echte Paarvergleichsdaten handelt, muss die Option `resptype` extra mit `resptype = "rating"` spezifiziert werden.

```
> despatt <- patt.design(issp2000, nitems = 6,
+     resptype = "rating")
> head(despatt)
```

```
    y CAR IND FARM WATER TEMP GENE u12 u13 u23 u14 u24
1 161   0   0    0     0    0    0   1   1   1   1   1
2  55   1   1    1     1    1   -5   1   1   1   1   1
3  22   1   1    1     1   -5    1   1   1   1   1   1
4  15   2   2    2     2   -4   -4   1   1   1   1   1
5   9   2   2    2     2   -3   -5   1   1   1   1   1
6   0   2   2    2     2   -5   -3   1   1   1   1   1
  u34 u15 u25 u35 u45 u16 u26 u36 u46 u56
1   1   1   1   1   1   1   1   1   1   1
2   1   1   1   1   1   0   0   0   0   0
3   1   0   0   0   0   1   1   1   1   0
4   1   0   0   0   0   0   0   0   0   1
5   1   0   0   0   0   0   0   0   0   0
6   1   0   0   0   0   0   0   0   0   0
```

Modellschätzung *Möglichkeit 2*

Um *einen* allgemeinen Effekt der Antwortkategorie „unentschieden" in diesem Pattern Modell für Ratingdaten berücksichtigen zu können, werden die 15 Kovariaten `u12, u13,` $\cdots$, `u56` (Spalte 8 bis 22) der γ-Parameter mit dem Befehl `rowSums()` zeilenweise summiert und im Objekt `U` temporär gespeichert.

```
> U <- rowSums(despatt[, 8:22])
> head(U)
```

```
[1] 15 10 10  7  6  6
```

Die Modellberechnung erfolgt dann mit folgendem Befehl:

```
> respatt2 <- gnm(y ~ CAR + IND + FARM + WATER +
+     TEMP + GENE + U, family = poisson, data = despatt)
> summary(respatt2)
Call:

gnm(formula = y ~ CAR + IND + FARM + WATER + TEMP + GENE + U,
    family = poisson, data = despatt)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-10.8374   -0.6202   -0.4768   -0.3585    8.3534

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.038530   0.046127 -65.873  < 2e-16 ***
CAR         -0.049508   0.012295  -4.027 5.66e-05 ***
IND          0.115271   0.012386   9.306  < 2e-16 ***
FARM         0.008470   0.012190   0.695 0.487183
WATER        0.041450   0.012197   3.398 0.000678 ***
TEMP         0.051543   0.012209   4.222 2.42e-05 ***
GENE         0.000000         NA      NA       NA
U            0.591194   0.005146 114.886  < 2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Std. Error is NA where coefficient has been constrained or is unidentified

Residual deviance: 2984.6 on 3956 degrees of freedom
AIC: 4196.5

Number of iterations: 5
```

Modellschätzung *Möglichkeit 3*

Die Designstruktur `despatt` wurde bereits generiert. Die Modellschätzung erfolgt wieder mittels der Funktion `gnm()`. Um nun für jeden der insgesamt 15 Paarvergleiche einen γ-Parameter, der die Antwortkategorie „unentschieden" repräsentiert, berechnen zu können, muss die Modellformel dementsprechend erweitert werden (mit `u12 + u13 + ⋯ + u56`).

```
> respatt3 <- gnm(y ~ CAR + IND + FARM + WATER +
+     TEMP + GENE + u12 + u13 + u23 + u14 +
+     u24 + u34 + u15 + u25 + u35 + u45 + u16 +
+     u26 + u36 + u46 + u56, family = poisson,
+     data = despatt)
> respatt3
Call:

gnm(formula = y ~ CAR + IND + FARM + WATER + TEMP + GENE + u12 +
    u13 + u23 + u14 + u24 + u34 + u15 + u25 + u35 + u45 + u16 +
    u26 + u36 + u46 + u56, family = poisson, data = despatt)


Coefficients:
(Intercept)          CAR          IND         FARM
  -3.239095    -0.064210     0.134930    -0.004887
      WATER         TEMP         GENE          u12
   0.038560     0.050831           NA     1.270165
        u13          u23          u14          u24
   0.365356     0.867787     0.307585     0.588040
        u34          u15          u25          u35
   1.194840     0.641683     0.638590     0.489905
        u45          u16          u26          u36
   0.644672    -0.111242     0.317241     0.690532
        u46          u56
   0.324214     0.772761

Deviance:            2588.049
Pearson chi-squared: 3545.954
Residual df:         3942
```

Ausgabe und Visualisierung der geschätzten Parameter

Im Folgenden wird die Ausgabe des berechneten γ-Parameters gezeigt.

```
> coef(respatt2)
Coefficients:
 (Intercept)          CAR          IND         FARM
-3.038530276 -0.049508426  0.115271345  0.008469744
       WATER         TEMP         GENE            U
 0.041450085  0.051543320           NA  0.591194307
```

Der $\hat{\gamma}$-Parameter befindet sich auf der 8. Position des Outputs der Funktion `coef()` und kann mit folgendem Befehl ausgegeben werden:

```
> U2 <- coef(respatt2)[8]
> U2
        U
0.5911943
```

Die $\hat{\gamma}_{(jk)}$-Parameter für einen bestimmten Paaarvergleich (jk) befinden sich auf den Positionen 8 bis 22 und werden ausgegeben mittels:

```
> u3 <- coef(respatt3)[8:22]
> u3
       u12        u13        u23        u14        u24
 1.2701647  0.3653563  0.8677874  0.3075853  0.5880404
       u34        u15        u25        u35        u45
 1.1948402  0.6416833  0.6385895  0.4899053  0.6446724
       u16        u26        u36        u46        u56
-0.1112420  0.3172411  0.6905320  0.3242136  0.7727610
```

Für die Ausgabe der geschätzten Objekt- und Werteparameter der Modelle `respatt2` und `respatt3` kann wieder die Funktion `patt.worth()` herangezogen werden.

```
> plambda2 <- patt.worth(respatt2, outmat = "lambda")
> plambda3 <- patt.worth(respatt3, outmat = "lambda")
```

und

```
> pworth2 <- patt.worth(respatt2)
> pworth3 <- patt.worth(respatt3)
```

Die Präferenz- bzw. Gefährlichkeitsskala mit den geschätzten Werteparametern wird einfach mit der `plot()`-Funktion erstellt.

```
> plot(pworth2)
```

bzw.

```
> plot(pworth3)
```