G. Rohwer, C. Dudel Wintersemester 2010/11

Methoden der Datenrepräsentation und Klassifikation

Kapitel 4: Eindimensionale Skalierung

4 Eindimensionale Skalierung

In gewisser Weise bildet die eindimensionale Skalierung nur einen Spezialfall der multidimensionalen Skalierung. Anstatt sich in erster Linie für räumliche Bilder zu interessieren, kann man sich jedoch im eindimensionalen Fall auch noch an zwei anderen Fragen orientieren. Erstens kann man sich auf die Frage beziehen, wie man unter Verwendung von Abstandsinformationen eine Menge von Objekten am besten in einer Reihenfolge anordnen kann. Diese Variante der Fragestellung erscheint insbesondere dann sinnvoll, wenn man aus theoretischen Gründen die Existenz einer Reihenfolge unterstellen kann; beispielweise in der Archäologie, wenn es darum geht, für eine Menge gefundener Artefakte eine zeitliche Reihenfolge zu bestimmen. Zweitens kann man eindimensionale Skalierung als eine Methode der Quantifizierung auffassen. Man versucht dann, die den Objekten durch das Skalierungsverfahren zugerechneten Zahlen als quantitativ interpretierbare Scores aufzufassen.

4.1 Methoden der Seriation

1. Unterschiedliche Problemformulierungen

Wir beziehen uns auf n Objekte, die durch eine Zahlenmenge $\mathcal{N} = \{1, \ldots, n\}$ repräsentiert werden und für die eine Abstandsmatrix $\mathbf{D} = (d_{ij})$ gegeben ist.

- In einer ersten Problemformulierung geht es nur darum, für die Objekte eine Reihenfolge zu bestimmen. Wir sprechen in diesem Fall von einem Problem der *nichtmetrischen eindimensionalen Skalierung* oder *Seriation*.[1]

[1] Der Ausdruck 'Seriation' wurde von D. G. Kendall (1971) eingeführt.

Tabelle 4.1-1 Werte von acht Variablen für 17 Typen chinesischer Bronzegefäße (`arch1.dat`). Quelle: Ihm (1978: 483), Elisseeff (1968: 109).

	Typ	Anzahl	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
1	A	14	1	1	1	1	1	1	1	1
2	B	1	1	1	1	0	1	1	1	1
3	C	5	1	0	1	1	1	1	1	1
4	D	18	1	0	0	1	1	1	1	1
5	F	1	1	1	1	1	0	1	1	1
6	H	1	1	0	1	1	0	1	1	1
7	J	11	1	0	0	1	0	1	1	1
8	K	1	1	0	0	0	0	1	1	1
9	M	1	1	0	0	0	0	0	1	1
10	N	14	0	0	0	1	0	1	0	0
11	P	1	0	0	0	1	0	0	0	0
12	R	6	1	0	0	0	1	1	1	1
13	S	1	1	0	0	0	1	0	1	1
14	T	1	1	0	0	0	1	0	1	0
15	V	32	1	0	0	0	1	1	0	0
16	X	2	1	0	0	0	1	0	0	0
17	Z	2	1	0	0	0	0	0	0	0

X_1 Position des Bügels: lateral (1), transversal (0)
X_2 Querschnitt des Bügels: gedreht (1), flach (0)
X_3 Bügelaufhängung: Ring (1), Maske (0)
X_4 Griff des Deckels: Knopf (1), Kuppel (0)
X_5 Kante: vorhanden (1), fehlernd (0)
X_6 Profil des Deckels: mit Hals (1), ohne Hals (0)
X_7 Ränder des Deckels: mit Vorsprung (1), ohne Vorsprung (0)
X_8 Form des Gefäßes: rund (1), unten ausgebaucht (0)

- In einer zweiten Problemformulierung geht es darum, korrespondierend zu den n Objekten reelle Zahlen $x_1, \ldots, x_n$ zu finden, so dass die euklidischen Abstände zwischen diesen Zahlen möglichst den vorgegebenen Abständen entsprechen, d.h. die Zahlen sollen aus einer Minimierung des folgenden Kriteriums gewonnen werden:

$$f(x_1, \ldots, x_n) = \sum\nolimits_{j<i} (d_{ij} - |x_i - x_j|)^2 \tag{4.1}$$

  Wir sprechen in diesem Fall von einem Problem der *metrischen eindimensionalen Skalierung*.

In diesem Abschnitt besprechen wir Methoden der Seriation, im nächsten Abschnitt die metrische eindimensionale Skalierung.

2. Ein Beispiel aus der Archäologie

Seriationsprobleme treten zum Beispiel in der Archäologie auf, wenn es sich darum handelt, Informationen über Ähnlichkeiten zwischen Artefakten zur

Tabelle 4.1-2 Aus den Daten in Tabelle 4.1-1 mit der Hamming-Distanz erzeugte Abstandsmatrix (`arch2.dat`).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	1	1	2	1	2	3	4	5	6	7	3	4	5	5	6	7
2	1	0	2	3	2	3	4	3	4	7	8	2	3	4	4	5	6
3	1	2	0	1	2	1	2	3	4	5	6	2	3	4	4	5	6
4	2	3	1	0	3	2	1	2	3	4	5	1	2	3	3	4	5
5	1	2	2	3	0	1	2	3	4	5	6	4	5	6	6	7	6
6	2	3	1	2	1	0	1	2	3	4	5	3	4	5	5	6	5
7	3	4	2	1	2	1	0	1	2	3	4	2	3	4	4	5	4
8	4	3	3	2	3	2	1	0	1	4	5	1	2	3	3	4	3
9	5	4	4	3	4	3	2	1	0	5	4	2	1	2	4	3	2
10	6	7	5	4	5	4	3	4	5	0	1	5	6	5	3	4	3
11	7	8	6	5	6	5	4	5	4	1	0	6	5	4	4	3	2
12	3	2	2	1	4	3	2	1	2	5	6	0	1	2	2	3	4
13	4	3	3	2	5	4	3	2	1	6	5	1	0	1	3	2	3
14	5	4	4	3	6	5	4	3	2	5	4	2	1	0	2	1	2
15	5	4	4	3	6	5	4	3	4	3	4	2	3	2	0	1	2
16	6	5	5	4	7	6	5	4	3	4	3	3	2	1	1	0	1
17	7	6	6	5	6	5	4	3	2	3	2	4	3	2	2	1	0

Begründung einer zeitlichen Reihenfolge auszunutzen.[2] Zur Illustration übernehmen wir ein Beispiel von P. Ihm (1978: 483), das auf Daten von V. Elisseeff (1968) beruht. Tabelle 4.1-1 zeigt die Daten.[3] Es handelt sich um chinesische Bronzegefäße, die in 17 Typen eingeteilt wurden. Für jeden Typ gibt es Werte von acht binären Variablen, deren Bedeutung am Ende der Tabelle angegeben ist.

Um aus den Daten eine Abstandsmatrix zu erzeugen, verwenden wir die Hamming-Distanz

$$d_{ij} := \sum_{k=1}^{8} |x_{ik} - x_{jk}|$$

durch die erfasst wird, in wievielen Variablen die Objekte i und j unterschiedliche Werte aufweisen. Tabelle 4.1-2 zeigt die Abstandsmatrix.[4]

Die Fragestellung lautet nun: Kann man gestützt auf diese Daten eine zeitliche Reihenfolge der Gefäßtypen bestimmen? Dabei soll die Annahme verwendet werden, dass es approximativ eine Entsprechung zwischen den Abständen in der Datenmatrix und den zeitlichen Abständen im Auftreten der Gefäßtypen gibt.

[2] Zur Diskussion von Seriationsproblemen in der Archäologie vgl. Laxton (1997).

[3] Ein Eintrag für den Typ Z wurde aufgrund der Angaben bei Elisseeff (1968: 109) verändert.

[4] Erzeugt mit dem Skript `arch2.cf`; das Datenfile mit der Abstandsmatrix wird `arch2.dat` genannt.

3. Kombinatorische Bestimmung einer Reihenfolge

In allgemeiner Form kann die Aufgabe folgendermaßen formuliert werden: Gesucht ist eine Permutation

$$\pi : \{1, \ldots, n\} \longrightarrow \{1, \ldots, n\}$$

so dass $\pi(i)$ die Position des Objekts i in der Reihenfolge angibt und die Entfernung der Objekte in der Reihenfolge möglichst gut den vorgegebenen Abständen entspricht. Die Entfernung von zwei Objekten i und j in der durch π gegebenen Reihenfolge kann durch

$$d_{ij}^{\pi} := |\pi(i) - \pi(j)|$$

erfasst werden. Unter Berücksichtigung der Möglichkeit, dass es in der Abstandsmatrix **D** Bindungen geben kann, liefern folgende Bedingungen ein Kriterium dafür, dass die Abstände in der Reihenfolge den vorgegebenen Abständen vollständig (nichtmetrisch) entsprechen:[5]

$$d_{ij} < d_{kl} \implies d_{ij}^{\pi} \leq d_{kl}^{\pi} \quad \text{und} \quad d_{ij} > d_{kl} \implies d_{ij}^{x} \geq d_{kl}^{\pi} \tag{4.2}$$

In den meisten Anwendungsfällen kann man natürlich nur erreichen, dass diese Bedingungen möglichst gut erfüllt werden, d.h. dass die Anzahl der Abstandsvergleiche, bei denen eine der Bedingungen verletzt wird, möglichst klein wird.

Um eine optimale Permutation zu finden, können kombinatorische Methoden verwendet werden; das wird im Anhang **??** näher erläutert. Für unser Beispiel verwenden wir zunächst die TDA-Prozedur `uds`, die eine Näherungslösung liefert.[6] Tabelle 4.1-3 zeigt das Ergebnis. Natürlich kann die zeitliche Richtung mit den Abständen allein nicht bestimmt werden.

Prüft man, wie gut in diesem Fall die Bedingungen (4.2) erfüllt werden, findet man, dass bei 668 von 9180 Abstandsvergleichen (etwa 7 %) eine der Bedingungen verletzt wird.

4. Nichtmetrische eindimensionale Skalierung

Anstelle kombinatorischer Verfahren können auch Programme für nichtmetrische MDS verwendet werden, wenn diese (was meistens möglich ist) auch eindimensionale Lösungen zulassen. Allerdings ist zu berücksichtigen, dass es im eindimensionalen Fall noch wesentlich schwieriger ist, ein globales Minimum der Stressfunktion zu finden.

[5] Es genügt, alle Elemente im unteren Dreieck der Abstandsmatrix zu betrachten. Also alle d_{ij} mit $i = 2, \ldots, n$ und $j = 1, \ldots, i-1$. Dann werden zu jedem dieser d_{ij}-Abstände alle d_{kl}-Abstände betrachtet, für die gilt: $k = i$ und $l = j, \ldots, k-1$ oder $k > i$ und $l = 0, \ldots, k-1$.

[6] Wir verwenden die Option 2, der ein von D. H. West (1983) entwickelter Algorithmus zur approximativen Lösung des QA-Problems zugrunde liegt. Für die Berechnung wurde das Skript `uds2.cf` verwendet.

Tabelle 4.1-3 Seriation der Abstandsmatrix in Tabelle 4.1-2 mit einer kombinatorischen Methode.

11	P
10	N
17	Z
16	X
15	V
14	T
9	M
13	S
8	K
12	R
7	J
4	D
6	H
3	C
2	B
1	A
5	F

Tabelle 4.1-4 Eindimensionale nichtmetrische Skalierungen der Abstandsmatrix in Tabelle 4.1-2 mit der TDA-Prozedur `mdsn`.

Bindungen: 1. Methode Stress: 0.1479			Bindungen: 2. Methode Stress: 0.2540		
1	A	-1.0170	5	F	-0.9897
5	F	-0.9507	1	A	-0.9570
2	B	-0.8584	2	B	-0.8697
3	C	-0.6780	3	C	-0.6457
6	H	-0.5724	6	H	-0.6088
4	D	-0.3549	4	D	-0.3394
7	J	-0.2463	7	J	-0.2940
12	R	-0.1439	12	R	-0.1561
8	K	-0.1172	8	K	-0.1065
9	M	0.0715	13	S	0.0734
13	S	0.0777	9	M	0.1249
14	T	0.4014	14	T	0.3783
15	V	0.5505	15	V	0.5445
16	X	0.7460	16	X	0.6997
17	Z	0.7861	17	Z	0.8336
10	N	1.0873	10	N	1.1024
11	P	1.2184	11	P	1.2100

Zur Illustration verwenden wir zunächst die TDA-Prozedur `mdsn`. Tabelle 4.1-4 zeigt die Ergebnisse für die beiden Möglichkeiten, um die Bindungen in der Abstandsmatrix zu berücksichtigen (vgl. Abschnitt 3.3, § 2). Die bei der zweiten Variante erzielte Reihenfolge entspricht derjenigen, die im vorangegangenen Paragraphen mit kombinatorischen Methoden erreicht wurde; bei der ersten Variante gibt es zwei Vertauschungen. Die Schwierigkeiten, ein globales Minimum zu finden, zeigen sich daran,

Box 4.1-1 Nichtmetrische eindimensionale MDS mit R.

```
> dat <- read.table("arch1.dat")
> rownames(dat) <- c("A","B","C","D","F","H","J",
                     "K","M","N","P","R","S","T","V","X","Z")

# Tabelle 4.1-2
> d <- dist(dat,method="manhattan")

> library(smacof)
> fit <- smacofSym(d,metric=F,ndim=1)
> fit2 <- metaSmacof(d,metric=F,ndim=1,runs=100)

> fit$stress.nm
[1] 0.02187130
> fit2
[1] 0.02187130

> order(fit$conf)
[1]  1  5  2  3  6  4  7 12  8  9 13 14 15 16 17 10 11
> order(fit2$conf)
[1]  1  5  2  3  6  4  7 12  8  9 13 14 15 16 17 10 11

> rownames(dat)[order(fit$conf)]
[1] "A" "F" "B" "C" "H" "D" "J" "R" "K" "M" "S" "T" "V" "X" "Z" "N" "P"

> fit$conf[order(fit$conf),]
          A           F           B           C           H           D
-1.00562544 -0.94047639 -0.84910958 -0.67034136 -0.56627773 -0.35084224
          J           R           K           M           S           T
-0.24379446 -0.14234989 -0.11587922  0.07104859  0.07667234  0.39682002
          V           X           Z           N           P
 0.54453908  0.73763344  0.77775979  1.07537677  1.20484628
```

dass bei jeweils 100 Wiederholungen (ausgehend von zufällig gewählten Anfangskonfigurationen) bei der ersten Methode nur in 6 Fällen, bei der zweiten Methode nur in einem Fall der relativ kleinste Stresswert erreicht wurde.

5. Berechnungen mit R

Der von West vorgeschlagene Algorithmus ist in `R` nicht implementiert, weshalb auf die bisher verwendeten Befehle der multidimensionalen Skalierung zurückgegriffen wird, die auch eine eindimensionale Skalierung erlauben.[7] Box 4.1-1 zeigt die Syntax für eine nichtmetrische eindimensionale Skalierung der Archäologiedaten.

Zunächst werden die Archäologiedaten geladen. Anschließend werden die Zeilen des Datensatzes entsprechend der Typenbezeichnung in Tabelle

[7] Eine Alternative stellt das Paket `seriation` dar, welches verschiedene Verfahren zur Seriation implementiert, die hier nicht weiter besprochen werden.

4.1-1 benannt und wird eine Abstandsmatrix wie in Tabelle 4.1-2 erzeugt. Zur Berechnung einer eindimensionalen Skalierung werden dann der Befehl `smacofSym` aus dem Paket `smacof` und die in Kapitel 3 definierte Funktion `metaSmacof` benutzt. Um festzulegen, dass das Ergebnis der Skalierung nicht wie bisher zwei Dimensionen aufweist, wird das Argument `ndim` auf den Wert `1` gesetzt. Vergleicht man die resultierenden Stresswerte der beiden erzeugten Anpassungen, zeigt sich, dass durch die Verwendung von 100 zufälligen Startkonfigurationen keine bessere Lösung erreicht werden konnte.

Die Reihenfolge der Typen ergibt sich aus der Reihenfolge der Punkte der Ergebniskonfiguration, die mit dem Befehl `order` angezeigt werden kann. Hierbei wird für jeden Wert aus `fit$conf` angegeben, welche Position dieser in der Reihenfolge hat, wobei diese mit dem niedrigsten Wert beginnt. Der erste Wert der Konfiguration, der zu Typus A gehört, ist der erste und somit kleinste Wert der Reihenfolge. Der zweite Wert der Konfiguration steht an fünfter Stelle der Reihenfolge, der dritte Wert an der zweiten Stelle und so fort.

Die mit dem Befehl `order` erzeugte Reihenfolge kann man nutzen, um die Bezeichnungen der Datenzeilen entsprechend zu sortieren. Hierfür wird der `order`-Aufruf in eckige Klammern hinter den Befehl `rownames` gesetzt. Der Aufruf von `rownames` zeigt die Zeilennamen des Datensatzes an, wobei durch die Angabe der Ordnung zunächst der erste Name ausgegeben wird, dann der fünfte und so weiter. Die gefundene Reihenfolge entspricht dem in Tabelle 4.1-4 dargestellten Ergebnis der ersten Methode.

Schließlich lässt sich relativ analog eine Sortierung der Punkte der Konfiguration entsprechend der errechneten Reihenfolge umsetzen. Hierfür wird hinter dem Aufruf der Konfiguration wieder der Befehl `order` in eckige Klammern gesetzt. Auf diesen folgt ein Komma, da die Ergebniskonfiguration in einer Matrix abgespeichert ist. Vergleicht man diese Werte mit denen in Tabelle 4.1-4, lässt sich eine relativ große Übereinstimmung feststellen.

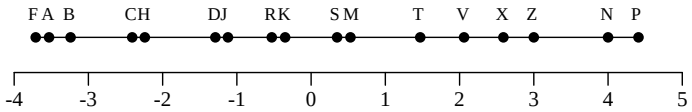
4.2 Metrische eindimensionale Skalierung

1. Illustration der Berechnung

In diesem Abschnitt besprechen wir die metrische eindimensionale Skalierung, bei der nicht nur eine Reihenfolge, sondern außerdem Positionen auf der Zahlengeraden bestimmt werden sollen. Mathematisch betrachtet geht es darum, Zahlen zu finden, die die Funktion (4.1) minimieren. Das Problem ist kompliziert, weil diese Funktion weder stetig differenzierbar noch global konvex ist (darauf gehen wir ausführlicher im Anhang **??** ein). Man muss deshalb Permutationsverfahren, (Gradienten-)Verfahren zur Funktionsminimierung und/oder Verfahren der linearen Programmie-

rung kombinieren.[8]

Für die praktische Durchführung der Berechnungen verwenden wir zunächst wieder die TDA-Prozedur `uds`. Für die metrische eindimensionale Skalierung beruht sie auf einem von Defays (1978) vorgeschlagenen Algorithmus, der durch eine Kombination unterschiedlicher Verfahren ein globales Minimum des Kriteriums (4.1) findet. Dieses Verfahren ist allerdings sehr rechenaufwendig, so dass es nur bis zu einer Anzahl von etwa 20 Objekten praktikabel ist. Für unser Beispiel liefert das Verfahren die in Tabelle 4.2-1 angegebene Lösung.[9] Die Reihenfolge ist offenbar identisch mit derjenigen, die durch das nichtmetrische Verfahren gefunden wurde. Zusätzlich erhält man jetzt zu jedem Objekt i einen Skalenwert x_i, der näherungsweise die Plazierung des Objekts auf der Zahlenachse angibt. Das folgende Bild veranschaulicht die Lage der Objekte auf der Zahlenachse.



2. Die Qualität der Skalierung

Mit den durch die metrische Skalierung erzeugten x-Werten kann offenbar eine neue Abstandsmatrix $\mathbf{D}^x = (d_{ij}^x)$ berechnet werden, wobei $d_{ij}^x := |x_i - x_j|$ ist. Der euklidische Abstand zwischen $\mathbf{D}$ und $\mathbf{D}^x$, also

$$\| \mathbf{D} - \mathbf{D}^x \| = \left(\sum\nolimits_{i \neq j} (d_{ij} - d_{ij}^x)^2 \right)^{1/2} \tag{4.3}$$

liefert dann ein Maß für die Qualität der Skalierung. In unserem Beispiel beträgt der Wert 16.79.

Vielleicht informativer ist jedoch die durchschnittliche absolute Differenz zwischen den vorgegebenen und den durch die Skalierung erzeugten Abständen, also

$$\frac{2}{n(n-1)} \sum\nolimits_{j<i} |d_{ij} - d_{ij}^x| \tag{4.4}$$

In unserem Beispiel findet man den Wert 0.81.

[8] Man vgl. hierzu die Beiträge von Defays (1978), Hubert und Arabie (1988), Pliner (1984, 1996), Lau, Leung und Tse (1998); Hubert, Arabie und Meulman (2002); Brusco (2002).

[9] Berechnet mit dem Skript `uds3.cf`.

Tabelle 4.2-1 Ergebnis einer metrischen eindimensionalen Skalierung der Abstandsmatrix in Tabelle 4.1-2.

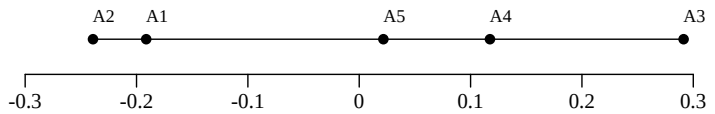
4.41	P
4.00	N
3.00	Z
2.59	X
2.06	V
1.47	T
0.53	M
0.35	S
-0.35	K
-0.53	R
-1.12	J
-1.29	D
-2.24	H
-2.41	C
-3.24	B
-3.53	A
-3.71	F

3. Skalierung der Klausuraufgaben

Als zweites Beispiel verwenden wir die Abstandsmatrix für die Klausuraufgaben (Tabelle 2.3-5 in Abschnitt 2.3). Die TDA-Prozedur `uds` findet in diesem Fall folgendes Ergebnis:

$A3$	0.2913
$A4$	0.1174
$A5$	0.0217
$A1$	-0.1913
$A2$	-0.2391

In graphischer Darstellung:



Die Reproduktion der ursprünglichen Abstandsmatrix gelingt jedoch nur schlecht; die entsprechend (4.4) berechnete durchschnittliche Differenz zwischen den ursprünglichen und den durch die Skalierung gewonnenen Abständen beträgt 0.081. Es wäre auch deshalb problematisch, die durch die Skalierung erzeugten Scores als Quantifizierungen des Schweregrads der Klausuraufgaben zu interpretieren.

Box 4.2-1 Metrische eindimensionale MDS mit Archäologiedaten.

```
> fit <- smacofSym(d,ndim=1)
> fit2 <- metaSmacof(d,ndim=1,runs=200)

> fit$stress.m
[1] 0.07094295
> fit2$stress.m
[1] 0.07082471

> rownames(dat)[order(fit$conf)]
[1] "A" "F" "B" "C" "H" "D" "J" "R" "K" "S" "M" "T" "V" "X" "Z" "N" "P"
> rownames(dat)[order(fit2$conf)]
[1] "F" "A" "B" "C" "H" "D" "J" "R" "K" "S" "M" "T" "V" "X" "Z" "N" "P"

> fit$conf[order(fit$conf),]
          A           F           B           C           H           D
-0.95342313 -0.93804534 -0.84577858 -0.63048949 -0.58435611 -0.33831143
          J           R           K           S           M           T
-0.29217806 -0.13840013 -0.09226675  0.09226675  0.13840013  0.38444481
          V           X           Z           N           P
 0.53822274  0.67662287  0.78426741  1.04568989  1.15333443
> fit$conf[order(fit2$conf),]
          F           A           B           C           H           D
-0.93804534 -0.95342313 -0.84577858 -0.63048949 -0.58435611 -0.33831143
          J           R           K           S           M           T
-0.29217806 -0.13840013 -0.09226675  0.09226675  0.13840013  0.38444481
          V           X           Z           N           P
 0.53822274  0.67662287  0.78426741  1.04568989  1.15333443

# Formel 4.3
> sqrt(sum((fit$obsdiss-fit$confdiss)^2))
[1] 3.134599

# Formel 4.4
> sum(abs(fit$obsdiss-fit$confdiss))*2/(17*16)
[1] 0.2125068

> sqrt(sum((fit2$obsdiss-fit2$confdiss)^2))
[1] 3.131937
> sum(abs(fit2$obsdiss-fit2$confdiss))*2/(17*16)
[1] 0.2113315
```

4. Berechnungen mit R

Für die Berechnungen mit R verwenden wir wieder den `smacofSym`-Befehl. Eine metrische eindimensionale Skalierung der Archäologiedaten findet sich in Box 4.2-1. Dabei wird davon ausgegangen, dass die Abstandsmatrix aus Tabelle 4.1-2 unter dem Namen `d` geladen ist. Dann kann analog zur nichtmetrischen MDS vorgegangen werden. Bei Verwendung von 200 zufälligen Startkonfigurationen erhält man eine nur geringfügig bessere Lösung und die beiden erzeugten Reihenfolgen ähneln sich relativ

Box 4.2-2 Metrische eindimensionale MDS mit Klausurdaten.

```
> dat <- matrix(c(39, 4, 0, 1, 2,
+    40, 1, 4, 0, 1,
+    25, 0, 2, 2,17,
+    21, 6, 9, 6, 4,
+    27, 6, 8, 0, 5),nrow=5,byrow=T)
> rownames(dat) <- c("A1","A2","A3","A4","A5")
> dat <- dat/rowSums(dat)
> d <- dist(dat,method="manhattan")*0.5

> fit <- smacofSym(d,ndim=1)
> fit2 <- metaSmacof(d,ndim=1,runs=200)

> fit$stress.m
[1] 0.1214350
> fit2$stress.m
[1] 0.08484305

> rownames(dat)[order(fit$conf)]
[1] "A2" "A1" "A5" "A3" "A4"
> rownames(dat)[order(fit2$conf)]
[1] "A2" "A1" "A5" "A4" "A3"

> sum(abs(fit$obsdiss-fit$confdiss))*2/(5*4)
[1] 0.2805696
> sum(abs(fit2$obsdiss-fit2$confdiss))*2/(5*4)
[1] 0.2391538
> mean(d)
[1] 0.3086957
```

stark, wobei die beiden ersten Elemente vertauscht sind. Die Lösung ohne zufällige Startkonfiguration entspricht der Lösung der nichtmetrischen Skalierung, während die zweite Reihenfolge mit der zweiten Lösung in Tabelle 4.1-4 und der Lösung in Tabelle 4.2-1 übereinstimmt.

Berechnet man zur Bewertung der Qualität der Anpassung die in (4.3) und (4.4) definierten Größen, findet man wieder, dass die beiden Reihenfolgen eine ähnlich gute Anpassung liefern. Dabei ist zu beachten, dass der Befehl `smacofSym` eine Skalierung der ursprünglichen Abstände vornimmt, so dass die hier für die beiden Größen angegebenen Werte nicht direkt mit denen aus § 2 vergleichbar sind.

Als zweites Beispiel verwenden wir wie in § 3 die Klausurdaten. Box 4.2-2 zeigt das R-Skript. Bei diesem Beispiel lässt sich durch die Verwendung zufälliger Startkonfigurationen eine bessere Lösung als im einfachen Fall finden. Hierauf verweisen sowohl der Wert der Stressfunktion als auch das Qualitätskriterium nach (4.4). Die aus der besseren Anpassung resultierende Reihenfolge entspricht der oben in § 3 gefundenen. Die für (4.4) gefundenen Werte zeigen auch, dass beide Anpassungen als relativ schlecht angesehen werden können, wenn man diese durchschnittlichen Differenzen

mit den ursprünglichen Abständen vergleicht.

5. Größere Mengen von Objekten

Das in § 1 verwendete Verfahren zur Minimierung des Kriteriums (4.1) ist nur praktikabel, wenn die Anzahl der Objekte klein ist (bis etwa $n = 20$). Bei größeren Anzahlen kann man folgende Möglichkeiten in Betracht ziehen. (a) Man kann zunächst mit dem in Abschnitt 4.1 (§ 3) besprochenen approximativen Verfahren eine näherungsweise optimale Reihenfolge ermitteln und dann innerhalb dieser Reihenfolge nach einem Minimum des Kriteriums (4.1) suchen. (b) Man kann ein für die multidimensionale Skalierung konzipiertes Verfahren verwenden. Hierbei stellt sich natürlich erneut das Problem, dass man normalerweise nur lokale Minima findet.