(7) Kanonische Korrelationsanalyse

Ziel dieser Sitzung ist es, die Durchführung einer kanonischen Korrelationsanalyse an einem Anwendungsbeispiel zu demonstrieren. In der kanonischen Korrelationsanalyse werden multivariate (d.h. mehrdimensionale) unabhängige und abhängige Variablen miteinander in Beziehung gesetzt, indem die am besten korrelierten Linearkombinationen dieser zwei Gruppen von Zufallsvariablen ermittelt werden.

Datensatz generieren

Wir beginnen damit, den im Folgenden verwendeten Datensatz im Rahmen einer Simulation zu erzeugen, um das der kanonischen Korrelationsanalyse zugrunde liegende generative Modell zu verstehen. Dafür werden zunächst unabhängige Variablen ("Prädiktoren") aus uniformen Verteilungen gezogen und zu einer Designmatrix zusammengefasst. Im Anschluss werden Regressionskoeffiezienten festgelegt, die von den Spalten der Designmatrix auf die Spalten der Datenmatrix abbilden. Schließlich wird mithilfe der multivariaten Normalverteilung eine Datenmatrix mit abhängigen Variablen ("Kriterien") im Sinne des multivariaten Allgemeinen Linearen Modells $Y = XB + E$, $\varepsilon_i^{\mathrm{T}} \sim N(0_{m_y}, I_{m_y})$ generiert, wobei $Y \in \mathbb{R}^{n \times m_y}$, $X \in \mathbb{R}^{n \times m_x}$, $B \in \mathbb{R}^{m_x \times m_y}$ und $\varepsilon_i \in \mathbb{R}^{1 \times m_y}$.

```r
# R-Pakete
library(MixMatrix)
set.seed(0)

# Datensimulation
n       = 20
m_x     = 2
m_y     = 2
x_1     = runif(n,10,30)
x_2     = runif(n, 0,10)
X       = matrix(c(x_1,x_2), nrow = n)
B       = matrix(c(1, 0.1,
                   1, 0.2),
                 nrow  = m_x,
                 byrow = T)
Y       = rmatrixnorm(n = 1, mean = X %*% B)
D       = data.frame(DUR  = X[,1],
                     EXP  = X[,2],
                     dBDI = Y[,1],
                     dGLU = Y[,2])

# Datenspeicherung
fname = "Kanonische_Korrelationsanalyse.csv"
write.csv(D, file = fname, row.names = FALSE)
```

Die resultierende Datenmatrix wird in einer CSV-Datei abgespeichert und kann beispielsweise in Tabellenform angezeigt werden.

```
D = read.csv("Kanonische_Korrelationsanalyse.csv")
knitr::kable(D,
    digits = 1,
    "pipe")
```

DUR	EXP	dBDI	dGLU
27.9	7.8	35.5	6.1
15.3	9.3	25.0	4.0
17.4	2.1	19.7	1.7
21.5	6.5	28.8	2.6
28.2	1.3	29.4	1.9
14.0	2.7	17.2	0.9
28.0	3.9	32.9	2.0
28.9	0.1	28.3	4.1
23.2	3.8	25.8	3.9
22.6	8.7	31.3	3.8
11.2	3.4	14.4	2.1
14.1	4.8	18.4	2.0
13.5	6.0	19.1	5.0
23.7	4.9	28.0	2.6
17.7	1.9	20.3	2.1
25.4	8.3	34.8	4.4
20.0	6.7	27.6	4.0
24.4	7.9	31.9	3.9
29.8	1.1	32.2	1.0
17.6	7.2	24.6	1.9

Univariate Korrelation

Im Anschluss erinnern wir zunächst an die Bestimmung eines Korrelationskoeffizienten bei univariaten (d.h. eindimensionalen) unabhängigen und abhängigen Variablen.

```
# Laden des Beispieldatensatzes
D     = read.csv("Kanonische_Korrelationsanalyse.csv")
x_i   = D$DUR
y_i   = D$dBDI
n     = length(x_i)

# "manuelle" Berechnung der Stichprobenkorrelation
x_bar = (1/n)*sum(x_i)
y_bar = (1/n)*sum(y_i)
s_x   = sqrt(1/(n-1)*sum((x_i - x_bar)^2))
s_y   = sqrt(1/(n-1)*sum((y_i - y_bar)^2))
c_xy  = 1/(n-1) * sum((x_i - x_bar) * (y_i - y_bar))
r_xy  = c_xy/(s_x * s_y)
print(r_xy)
```

```
[1] 0.8829308
```

Die gleiche Analyse können wir auch im Sinne eines Black-Box-Verfahrens mit der **R**-Funktion `cor()` durchführen.

```
# Automatische Berechnung mit cor()
r_xy  = cor(x_i, y_i)
print(r_xy)
```

```
[1] 0.8829308
```

Multivariate Korrelation

Wir führen schließlich eine kanonische Korrelationsanalyse zu dem in der Vorlesung betrachtetem Anwendungsbeispiel durch.

```
# R-Paket
library(expm)

# Datenpräprozessierung
D           = read.csv("Kanonische_Korrelationsanalyse.csv")
x           = as.matrix(cbind(D$DUR , D$EXP))
y           = as.matrix(cbind(D$dBDI, D$dGLU))
n           = nrow(x)
m_x         = ncol(x)
m_y         = ncol(y)
Y           = t(cbind(x,y))

# Stichprobenkovarianzmatrixpartition
I_n         = diag(n)
J_n         = matrix(rep(1,n^2), nrow = n)
C           = (1/(n-1))*(Y %*% (I_n-(1/n)*J_n) %*% t(Y))
C_xx        = C[1:m_x,1:m_x]
C_xy        = C[1:m_x,(m_x+1):(m_x+m_y)]
C_yx        = C[(m_x+1):(m_x+m_y),1:m_x]
C_yy        = C[(m_x+1):(m_x+m_y),(m_x+1):(m_x+m_y)]

# Kanonische Korrelationsanalyse
K_hat       = sqrtm(solve(C_xx)) %*% C_xy %*% sqrtm(solve(C_yy))
ALB_hat     = svd(K_hat)
A_hat       = ALB_hat$u
Lambda_hat  = ALB_hat$d
B_hat       = ALB_hat$v
a_hat       = sqrtm(solve(C_xx)) %*% A_hat
b_hat       = sqrtm(solve(C_yy)) %*% B_hat
rho_hat     = as.matrix(Lambda_hat)

# Ausgabe
cat(  "rho_hat_1 : ", rho_hat[1],
    "\na_hat_1   : ", a_hat[,1],
    "\nb_hat_1   : ", b_hat[,1],
    "\nrho_hat_2 : ", rho_hat[2],
    "\na_hat_2   : ", a_hat[,2],
    "\nb_hat_2   : ", b_hat[,2], "\n")
```

```
rho_hat_1 :  0.9950575
a_hat_1   :  -0.1623409 -0.173979
b_hat_1   :  -0.1554175 -0.05025419
rho_hat_2 :  0.5010358
a_hat_2   :  -0.06026274 0.3118808
b_hat_2   :  -0.08128072 0.7773036
```

Die gleiche Analyse können wir auch im Sinne eines Black-Box-Verfahrens mit der **R**-Funktion `cancor()` durchführen.

```
# Datenpräprozessierung
D           = read.csv("Kanonische_Korrelationsanalyse.csv")
x           = as.matrix(cbind(D$DUR , D$EXP))
y           = as.matrix(cbind(D$dBDI, D$dGLU))
cca         = cancor(x,y)

# Ausgabe
cat(  "rho_hat_1 : ", cca$cor[1],
    "\nrho_hat_2 : ", cca$cor[2], "\n")
```

```
rho_hat_1 :  0.9950575
rho_hat_2 :  0.5010358
```

Im vorliegen Fall würden also die Linearkombination $\xi = 0.16\ \text{DUR}\ + 0.17\ \text{EXP}\ $ als "bester Prädiktor" und die Linearkombination $\upsilon = 0.15\ \text{dBDI} + 0.05\ \text{dGLU}\ $ als "am besten prädizierbares Kriterium" die kanonische Korrelation maximieren und der kanonische Korrelationskoeffizient von 0.99 signalisiert eine hochgradige multivariate Korrelation.