# Skriptum zur Vorlesung Wahrscheinlichkeitstheorie und Statistik (Anhang Matlab)

**Robert Eberle, Thomas Fetz**

2022S 844263/844264 VO Wahrscheinlichkeitstheorie und Statistik für Elektrotechnik/Mechatronik (VU / 2h / 2,5 ECTS-AP)

# Inhaltsverzeichnis

# Anhang A

# MATLAB für Wahrscheinlichkeitstheorie und Statistik

In diesem Anhang zum Skriptum werden für die einzelnen Kapitel die benötigten MATLAB-Befehle vorgestellt.

Hinweise zur Nutzung:

- Die Nummerierung der einzelnen Abschnitte entspricht der Kapitelnummerierung im Skriptum.
- Die hier verwendeten Dateien (*.m, *.txt, *.xlsx) finden Sie im Olat im Ordner MATLAB.
- Der hier vorgestellte Programmcode kann einfach in das Kommando-Fenster von MATLAB kopiert und ausgeführt werden.
- Manchmal benötigen Sie dazu auch den zuvor angeführten Programmcode, damit es funktioniert.
- Manchmal ist es auch nützlich, sich die Dateien mit den Daten anzuschauen, um die Programme zu verstehen.

## A.2 Ad Kapitel 2

### A.2.1 Importieren von Daten

#### A.2.1.1 Aus Text-Dateien

Enthält eine Text-Datei nur eine einzelne Matrix, kann diese mit `load` gelesen werden. In der Datei `wetter.txt` befindet sich eine Matrix, wobei die erste Spalte die Tage im Jänner von 1 bis 31 enthält und die zweite Spalte die Tagesmittelwerte der Temperatur in Innsbruck aus dem Jahr 2001. Die eingelesene Matrix `D` zerlegen wir in einen Vektor `Tag` und in einen Vektor `T` für die Temperatur.

```
D = load('wetter.txt');
Tag = D(:,1);  % zerlegen in Tag und T
T = D(:,2);
```

Alternativ können wir diese Datei auch mit `readtable` lesen. Das erhaltene Objekt `Table` enthält dann die Eigenschaften `Var1` und `Var2` (die beiden Spalten) sowie `Variables` (die gesamte Matrix).

```
Table = readtable('wetter.txt');
Tag = Table.Var1;
T = Table.Var2;
% oder
D = Table.Variables;
Tag = D(:,1); % zerlegen in Tag und T
T = D(:,2);
```

Falls wir `Table` ausgeben (; weglassen) erhalten wir

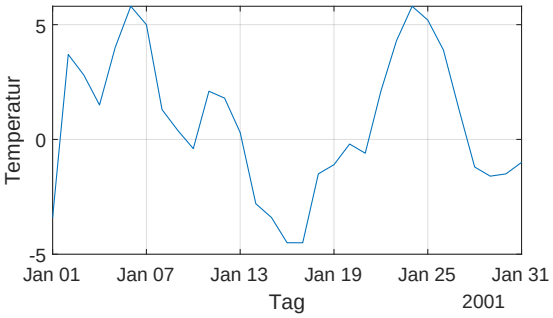
```
Table =
31x2 table
Var1    Var2
----    ----
  1     -3.4
  2      3.7
  3      2.8
  4      1.5
  5        4
  6      5.8
u.s.w.
```

Bei Verwendung von `readtable` darf auch ein Datum und eine Kopfzeile vorkommen (ein Wort pro Spalte), z.B.:

```
Tag         Temperatur
01.01.2001  -3,4
02.01.2001   3,7
03.01.2001   2,8
04.01.2001   1,5
05.01.2001   4
```

Dann heißen die Eigenschaften in `Table` wie die Worte in der Kopfzeile (keine Sonderzeichen!) und es funktioniert auch mit einem Komma statt einem Dezimalpunkt, falls man dies angibt. Die Eigenschaft `Table.Variables` funktioniert dann aber nicht mehr, da das Datum ein String und die Temperatur eine Zahl ist, was nicht zu einer Matrix zusammengefügt werden kann. Wir plotten auch noch die Temperaturkurve.

```
Table = readtable('wetter2.txt','decimal',',');
Tag = Table.Tag;    % zerlegen in Tag und T
T = Table.Temperatur;
plot(Tag,T), xlabel('Tag'), ylabel('Temperatur'), grid on
```




| | A | B |
|---|---|---|
| 1 | Tag | Temperatur |
| 2 | 01.01.2001 | -3,4 |
| 3 | 02.01.2001 | 3,7 |
| 4 | 03.01.2001 | 2,8 |
| 5 | 04.01.2001 | 1,5 |
| 6 | 05.01.2001 | 4 |
| 7 | 06.01.2001 | 5,8 |
| 8 | 07.01.2001 | 5 |

Wetter 2001 Wetter 2002

Abb. A.1: Temperaturkurve mit Datum, Ausschnitt aus Excel-Datei `wetter.xlsx`.

#### A.2.1.2 Aus Excel-Dateien

Mit `readtable` können auch aus Excel-Dateien Daten importiert werden. Ein Tabellenblatt darf nur eine einzige Tabelle enthalten und kann mit `'Sheet','Tabellenblatt'` ausgewählt werden.

```
%                  Excel-Datei   Tabellenblatt
Table = readtable('wetter.xlsx','Sheet','Wetter 2001');
Tag = Table.Tag;  % zerlegen in Tag und T
T = Table.Temperatur;
plot(Tag,T), xlabel('Tag'), ylabel('Temperatur'), grid on
```

### A.2.2 Graphische Darstellung von Daten

#### A.2.2.1 Säulendiagramme

Säulendiagramme werden für diskrete Daten verwendet, z.B. für die Schulnoten aus der Vorlesung. Wir lesen aus der Datei `Noten.xlsx` das Tabellenblatt `Noten Mathematik`. Die Häufigkeiten erhalten wir aus der Spalte `Mathematik` und die "Kategorien" (hier Noten) aus den Zeilennamen (`Row`). Achtung: Falls die Zeilennamen auch einen Spaltennamen besitzen, so findet man die Kategorien nur unter dem Spaltennamen. Damit das Histogramm auch wirklich ein Säulendiagramm ist, reduzieren wir die Breite `BarWidth` der Balken (Säulen) auf 50%. Die absoluten Häufigkeiten erhalten wir mit:

```
Noten=readtable('Noten.xlsx','Sheet','Noten␣Mathematik','ReadRowNames',true);
C = Noten.Row; % Zeilennamen bzw. Kategorien, hier Noten
H = Noten.Mathematik; % Häufigkeiten in Spalte Mathematik
histogram('Categories',C,'BinCounts',H,'BarWidth',.5)
grid on, ylabel('abs.␣Häufigkeiten');
title('Säulendiagramm,␣abs.␣Häufigkeiten')
```

Und die relativen Häufigkeiten mit `'Normalization','probability'` als weitere Argumente:

```
histogram('Categories',C,'BinCounts',H,'BarWidth',.5,...
            'Normalization','probability')
grid on, ylabel('rel.␣Häufigkeiten');
title('Säulendiagramm,␣rel.␣Häufigkeiten')
```

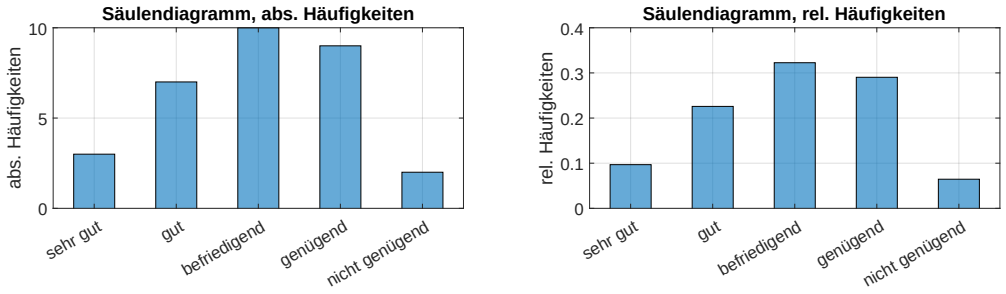



Abb. A.2: Säulendiagramme mit absoluten und relativen Häufigkeiten.

Im Tabellenblatt `Noten Mathematik Englisch` sind Mathematik- und Englischnoten als Liste (nicht die Häufigkeiten) enthalten. Wir möchten ein Säulendiagramm für die Mathematiknoten zeichnen. Dazu müssen die Noten als Zahlen in Kategorien umgewandelt werden.

Einfache Variante:

```
Noten = readtable('Noten.xlsx','Sheet','Noten␣Mathematik␣Englisch');
L = Noten.Mathematik; % Notenliste in Spalte Mathematik
histogram(categorical(L),'BarWidth',.5)
grid on; ylabel('abs.␣Häufigkeiten');
```

Mit Übersetzung der Ziffernnoten in ’Einser’, ’Zweier’, u.s.w.

```
Noten = readtable('Noten.xlsx','Sheet','Noten␣Mathematik␣Englisch');
L = Noten.Mathematik; % Notenliste in Spalte Mathematik
%           Daten, Übersetzung der Tabellenwerte in Text
C = categorical(L,[1 2 3 4 5],{'Einser','Zweier','Dreier','Vierer','Fünfer'});
histogram(C,'BarWidth',.5)
grid on; ylabel('abs.␣Häufigkeiten');
```

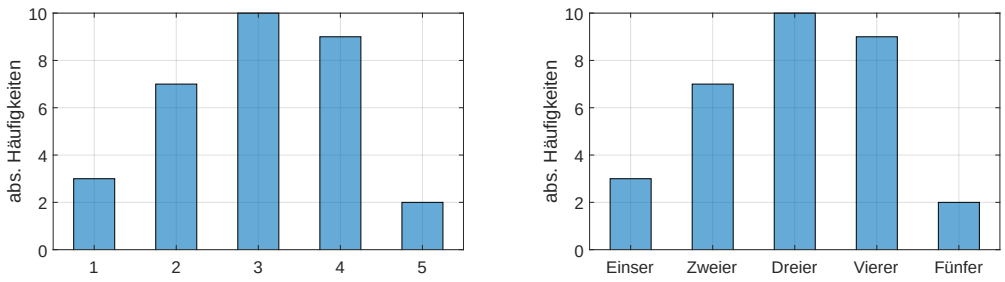



Abb. A.3: Säulendiagramme aus Kategorien.

Für kontinuierliche Daten werden Säulendiagramme oder Histogramme verwendet. Als Daten nehmen wir wieder die Jänner-Temperaturen in Innsbruck aus dem Jahr 2001 und die Klassen $[-7,-5),[-5,-3),\ldots,[5,7]$. Die Klassengrenzen `K` werden mit dem Vektor $[-7,-5,\ldots,7]$ festgelegt.

```
Daten_2001 = readtable('wetter.xlsx','Sheet','Wetter 2001');
Tag = Daten_2001.Tag;
T = Daten_2001.Temperatur;
K = -8:2:8;
```

Wir können für die Höhe der Säulen die absoluten Häufigkeit (Säulendiagramm) verwenden:

```
histogram(T,K,'FaceColor','y') % Farbe gelb
xlabel('Temperatur'), ylabel('absolut Häufigkeit')
title('Säulendiagramm, abs. Häufigkeit'), grid on
```

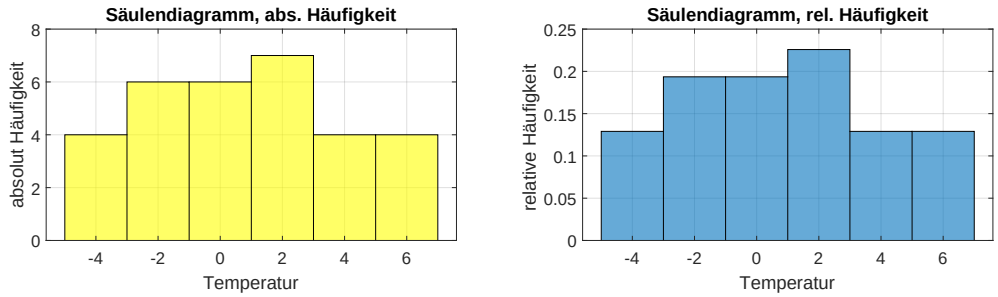



Abb. A.4: Säulendiagramme mit absoluten und relativen Häufigkeiten.

Oder die relative Häufigkeit (auch Säulendiagramm):

```
histogram(T,K,'Normalization','probability')
xlabel('Temperatur'), ylabel('relative Häufigkeit ')
title('Säulendiagramm, rel. Häufigkeit'), grid on
```

Leider kann die Säulenbreite nur eingestellt werden, falls Kategorien, nicht aber Klassengrenzen übergeben werden. Wir bemerken noch, dass wir im Skriptum für die Klassengrenzen $(a,b]$ und hier $[a,b)$, wie in `histogram` definiert, verwenden, was zu unterschiedlichen Resultaten führt. Für Grenzen wie im Skriptum, müsste man "selber zählen" und die Eigenschaft `'BinCounts'` verwenden.

### A.2.2.2 Histogramme

Für ein echtes Histogramm, wie in der Vorlesung definiert, wird eine flächentreue Darstellung (Gesamtfläche der Balken ist 1, Wahrscheinlichkeitsdichte) verwendet, d.h. als Höhe wird die relative Häufigkeit dividiert durch die Klassenbreite genommen. Dazu sind in `histogram` die Argumente `'Normalization','pdf'` anzugeben.

```
histogram(T,K,'Normalization','pdf')
xlabel('Temperatur'), ylabel('rel. Häufigkeit (flächentreu)')
title('Histogramm lt. Skriptum'), grid on
```

Es kann auch die empirische Verteilungsfunktion entweder als Balken oder Kurven (als Treppenfunktion oder als Kurve durch die Balkenmittelpunkte) dargestellt werden. Die Balkenhöhen erhält man über die Eigenschaft `Values`.

```
% empirische Verteilungsfunktion als Balken
h = histogram(T,K,'Normalization','cdf');
hold on
% als Treppenfunktion
stairs([K(1),K],[0,h.Values,h.Values(end)],'r','linewidth',2)
% durch die Balkenmittelpunkte
m = (K(1:end-1)+K(2:end))/2;
plot([K(1),m,K(end)],[0,h.Values,1],'b','linewidth',2)
xlabel('Temperatur'), ylabel('Wahrscheinlichkeit')
title('empirische Verteilungsfunktion'), grid on
```

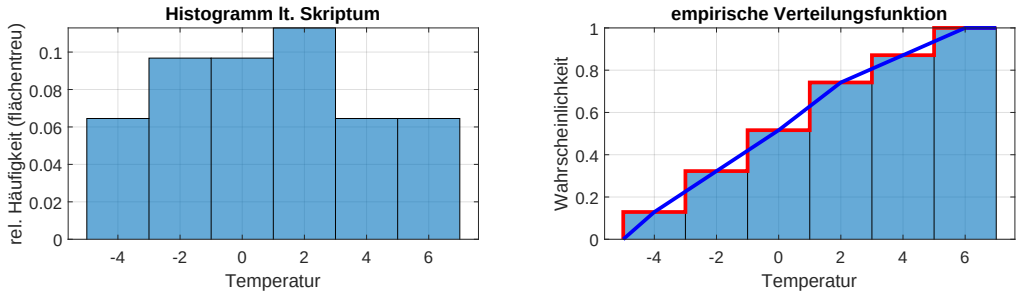



Abb. A.5: Flächentreues Histogramm und empirische Verteilung als Balken oder Kurven.

box

Es sind auch unterschiedliche Klassenbreiten möglich, was in Excel nur schwierig umzusetzen wäre:

```
K = [-8,-4,-2,0,2,4,8];
histogram(T,K,'Normalization','pdf')
xlabel('Temperatur'), ylabel('rel. Häufigkeit (flächentreu)')
title('Histogramm lt. Skriptum'), grid on
```

Es lassen sich auch zwei Histogramme überlagern, z.B. für die Temperaturen 2001 und 2002, was aber für dieses Beispiel nicht so gut aussieht:

```
Daten_2001 = readtable('wetter.xlsx','Sheet','Wetter 2001');
Tag = Daten_2001.Tag;
T = Daten_2001.Temperatur;
K = -7:2:7;
histogram(T,K,'Normalization','pdf')
Daten_2002 = readtable('wetter.xlsx','Sheet','Wetter 2002');
T = Daten_2002.Temperatur;
hold on
histogram(T,K,'Normalization','pdf')
xlabel('Temperatur'), ylabel('rel. Häufigkeit (flächentreu)')
title('Überlagerung'), grid on
legend('2001','2002')
```

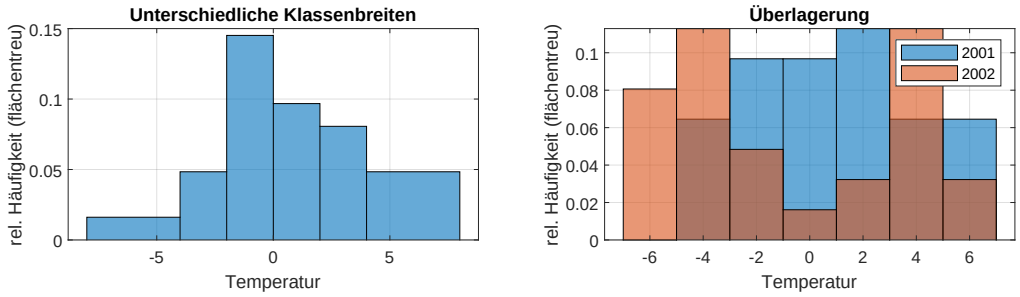



Abb. A.6: Unterschiedliche Klassenbreiten und Überlagerung von zwei Histogrammen.

### A.2.2.3 Boxplot

Einen Boxplot erhalten wir einfach mit dem Befehl `boxplot`. Es können auch mehrere Boxplots zum Vergleich nebeneinander gezeichnet werden, z.B. für die Temperaturen von 2001 und 2002. Dazu müssen die Daten zu einer Matrix zusammengefasst werden. Das optionale Cell-Array mit den Jahren dient zur Beschriftung:

```
Daten_2001 = readtable('wetter.xlsx','Sheet','Wetter 2001');
Daten_2002 = readtable('wetter.xlsx','Sheet','Wetter 2002');
T = [Daten_2001.Temperatur,Daten_2002.Temperatur];
boxplot(T,{'2001','2002'})
```

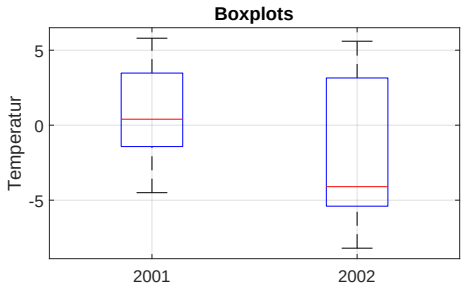



Abb. A.7: Zwei Boxplots nebeneinander.

### A.2.3 Mittelwert, Median, Modalwert, Standardabweichung und Varianz, Quantile

Diese Werte werden mit den Befehlen `mean`, `median`, `mode` (häufigster Wert), `std` und `var` berechnet. Ist das Argument eine Matrix, so wird die Berechnung für jede Spalte durchgeführt. Wir verwenden als Beispiel die Jänner-Temperaturen für 2001 sowie 2002 und fassen dazu beide Vektoren zu einer Matrix zusammen. Damit erhalten wir alle Werte für 2001 und 2002.

```
Daten_2001 = readtable('wetter.xlsx','Sheet','Wetter␣2001');
Daten_2002 = readtable('wetter.xlsx','Sheet','Wetter␣2002');
T = [Daten_2001.Temperatur,Daten_2002.Temperatur];
Mittelwert = mean(T)
Median = median(T)
Modalwert = mode(T)
Standardabweichung = std(T)
Varianz = var(T)

Mittelwert = 0.7613   -1.7968
Median =     0.4000   -4.1000
Modalwert = -4.5000   -7.1000
Standardabweichung = 3.0180    4.5788
Varianz =    9.1085   20.9657
```

Die 5%-, 10%-, 50%-, 90%- und 95%-Quantile für die Jänner-Temperaturen für 2001 und 2002 erhalten wir folgendermaßen:

```
p = [0.05 0.1 0.5 0.9 0.95]';
Q = quantile(T,p);
[p,Q] =
0.0500   -4.4450   -7.7650
0.1000   -3.4000   -7.1000
0.5000    0.4000   -4.1000
0.9000    5.0800    4.3000
0.9500    5.7700    5.1550
```

Achtung: Leider gibt viele unterschiedliche Definitionen, die Quantilwerte zu berechnen, was zu unterschiedlichen Resultaten führen kann. Nur das 50%-Quantil (Median) ist immer gleich.

Wir passen noch an die Temperaturdaten von 2001 eine Normalverteilung an, indem wir Mittelwert und Varianz wie oben bestimmen. Zum Zeichnen der Dichtefunktion der Normalverteilung verwenden wir `pdf('Normal',x,mu,sigma)`.

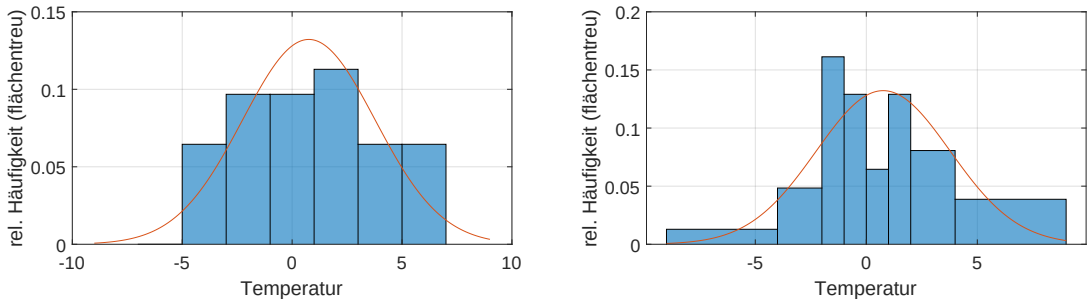



Abb. A.8: Histogramme mit angepasster Dichte einer Normalverteilung, unterschiedliche Klassen und Klassenbreiten.

```
Daten_2001 = readtable('wetter.xlsx','Sheet','Wetter 2001');
Tag = Daten_2001.Tag;
T = Daten_2001.Temperatur;
mu = mean(T); sigma = std(T);
K = -7:2:7 % oder
K = [-9 -4 -2 -1  0 1 2 4 9];
histogram(T,K,'Normalization','pdf')
hold on, t = linspace(-9,9,200);
plot(t,pdf('Normal',t,mu,sigma)), grid on
xlabel('Temperatur'), ylabel('rel. Häufigkeit (flächentreu)')
```

## A.3 Ad Kapitel 3

### A.3.1 Graphische Darstellung

#### A.3.1.1 Streudiagramm

Wir stellen die Temperaturen und die Bedeckung aus dem Jahr 2002 gemeinsam als Streudiagramm (Scatterplot) dar. Dazu zeichnen wir einfach mit `plot` die Datenpunkte.

```
Daten_2002 = readtable('wetter.xlsx','Sheet','Wetter 2002');
D = [Daten_2002.Temperatur,Daten_2002.Bedeckung];
plot(D(:,1),D(:,2),'o')
xlabel('Temperatur'), ylabel('Bedeckung')
title('Streudiagramm'), grid on
```

#### A.3.1.2 Heatmap

Für die Schulnoten in Mathematik und Englisch ist ein Streudiagramm nicht geeignet, da dort viele Punkte mehrfach vorkommen und diese Information im Streudiagramm verloren gehen würde. Wir verwenden dafür eine Heatmap, die für die Häufigkeiten unterschiedliche Farben verwendet. Dazu müssen dir die Tabelle `Noten` und die Spaltennamen `'Mathematik'` bzw. `'Enlisch'` übergeben.

```
Noten = readtable('Noten.xlsx','Sheet','Noten Mathematik Englisch');
heatmap(Noten,'Mathematik','Englisch')
xlabel('Englisch'), ylabel('Mathematik')
title('Heatmap')
```

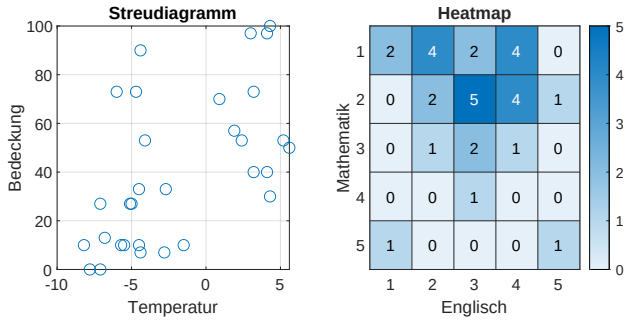



Abb. A.9: Streudiagramm für Temperatur und Bedeckung sowie eine Heatmap für die Schulnoten in Mathematik und Englisch.

### A.3.2 Kovarianzmatrix und Korrelationskoeffizienten

Wir berechnen für die Temperatur und Bedeckung aus 2002 die Kovarianzmatrix und die Pearson-Korrelationskoeffizienten sowie die Spearman-Rangkorrelationskoeffizienten als Matrizen.

```
Daten_2002 = readtable('wetter.xlsx','Sheet','Wetter 2002');
D = [Daten_2002.Temperatur,Daten_2002.Bedeckung];
Cov = cov(D)
R = corr(D) % oder corr(D,'Type','Pearson')
rho = corr(D,'Type','Spearman')

Cov =
1.0e+02 *
0.209656559139785   0.808497849462365
0.808497849462365   9.645956989247312

R =
1.000000000000000   0.568528161009985
0.568528161009985   1.000000000000000
rho =
1.000000000000000   0.570216436182175
0.570216436182175   1.000000000000000
```

Falls wir im Schulnotenbeispiel bereits von den Häufigkeiten ausgehen (Tabelle wie im Skriptum) statt von den reinen Daten, dann müssen wir etwas mehr tun, um $S_{xy}$ und $R$ zu erhalten:

```
Noten = readtable('Noten.xlsx','Sheet','Noten␣2D');
M = Noten.Variables;
H = M(2:end,2:end);
Hx = sum(H,2); Hy = sum(H,1); n = sum(H,'all');
x = M(2:end,1); y = M(1,2:end);
mx = sum(x.*Hx)/n; my = sum(y.*Hy)/n;
sx = sqrt(sum((x-mx).^2.*Hx)/(n-1));
sy = sqrt(sum((y-my).^2.*Hy)/(n-1));
sxy = sum(H.*(x-mx).*(y-my),'all')/(n-1)
R = sxy/(sx*sy)

sxy =
0.133333333333333
R =
0.108147614087175
```

Wir verwenden dabei, dass `v.*w` das Tensorprodukt $v \otimes w$ ist, falls $v$ eine Spalte und $w$ eine Zeile ist. Daher ergibt `(x-mx).*(y-my)` eine Matrix mit allen Produkten, die dann mit allen Häufigkeiten in `H` elementweise multipliziert wird und deren Elemente mit `sum(...,'all')` aufsummiert werden.

### A.3.3 Ausgabe von Tabellen

Mit `writetable` können Tabellen in Excel-Dateien geschrieben werden. Eine schöne formatierte Ausgabe der Korrelationskoeffizienten `R` von weiter oben erhalten wir im Kommandofenster mit `fprintf`

```
fprintf('␣␣␣␣␣␣␣␣␣␣␣Temperatur␣␣␣Bedeckung\n');
fprintf('Temperatur␣%10.6f␣␣%10.6f\n',R(1,:));
fprintf('Bedeckung␣␣%10.6f␣␣%10.6f\n',R(2,:));

           Temperatur   Bedeckung
Temperatur   1.000000    0.568528
Bedeckung    0.568528    1.000000
```

oder mit `table`:

```
T = array2table(R);
T.Properties.VariableNames = {'Temperatur','Bedeckung'};
T.Properties.RowNames = T.Properties.VariableNames; T % Objekt T ausgeben

T =
2x2 table
            Temperatur     Bedeckung
            ----------     ---------
Temperatur           1       0.56853
Bedeckung      0.56853             1
```