Statistiksoftware R

Lektion B05 – BSc Wahrscheinlichkeit und Statistik

Digital Finance

Gliederung

Lernziele

Am Ende dieser Lektion werden Sie in der Lage sein:

1. Die R-Konsole zu benutzen und Variablen, Vektoren und Funktionsaufrufe zu verstehen
2. Diagramme (Barplot, Histogramm, Boxplot, ECDF, Streudiagramm) in R zu erstellen und zu interpretieren
3. Das d/p/q/r-Schema für Verteilungen in R zu erklären und anzuwenden
4. Wahrscheinlichkeiten diskreter Verteilungen (Binomial, Poisson) mit R zu berechnen
5. Wahrscheinlichkeiten und Quantile stetiger Verteilungen (Normal, Uniform) mit R zu berechnen
6. Verteilungen grafisch darzustellen und zu vergleichen

Diese Ziele leiten, was Sie aus dieser Lektion beherrschen sollten.

Die R-Konsole

R ist eine freie Programmiersprache für statistische Berechnungen und Grafiken.

Rechnen und Zuweisen:

```
# Einfache Berechnungen
2 + 3          # 5
sqrt(16)       # 4
log(100)       # 4.6052 (nat. Log.)

# Variablen zuweisen
x <- 5
y <- 3
x + y          # 8
```

Hilfe aufrufen:

```
# Hilfe zu einer Funktion
?mean
help(sd)

# Beispiele anzeigen
example(hist)
```

Wichtig: Der Zuweisungsoperator in R ist <−, nicht =.

RStudio bietet eine komfortable Oberfläche für R mit Editor, Konsole, Plots und Hilfe.

Vektoren und grundlegende Funktionen

Vektoren sind das zentrale Datenobjekt in R.

Vektoren erstellen:

```r
# Vektor mit c()
renditen <- c(0.02, -0.01, 0.03,
              -0.005, 0.015)

# Sequenzen
1:10          # 1, 2, ..., 10
seq(0, 1, by=0.1)
```

Statistische Funktionen:

```r
mean(renditen)     # Mittelwert
sd(renditen)       # Standardabw.
median(renditen)   # Median
summary(renditen)  # Zusammenfassung
length(renditen)   # Anzahl: 5
```

```r
# Vektorisierte Operationen
renditen * 100               # Alle Werte * 100
cumsum(renditen)             # Kumulative Summe
renditen[renditen > 0]       # Nur positive Renditen
```

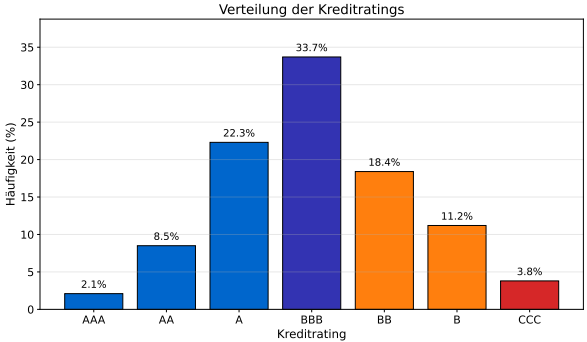
R arbeitet vektorisiert – Schleifen sind selten nötig.

Barplot: Kategorische Daten

barplot() stellt Häufigkeiten kategorischer Variablen dar.

```r
# Kreditrating-Verteilung
ratings <- c(AAA=2.1, AA=8.5,
             A=22.3, BBB=33.7,
             BB=18.4, B=11.2,
             CCC=3.8)

barplot(ratings,
  main="Kreditratings",
  ylab="Haeufigkeit (%)",
  col="steelblue",
  border="black")
```



Benannte Vektoren erzeugen automatisch Achsenbeschriftungen.

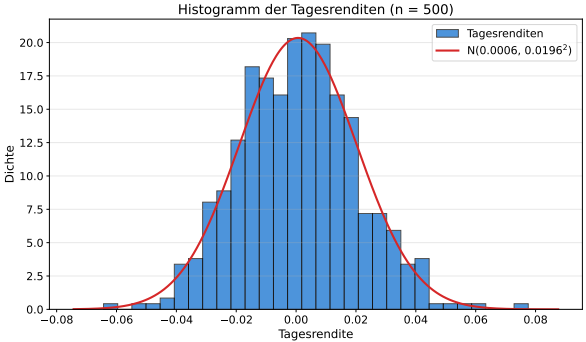
Histogramm: Stetige Daten

hist() zeigt die Verteilung stetiger Variablen.

```r
# Tagesrenditen simulieren
set.seed(42)
r <- rnorm(500,
           mean=0.0005,
           sd=0.02)

hist(r, breaks=30,
  freq=FALSE,
  main="Tagesrenditen",
  xlab="Rendite",
  col="steelblue",
  border="black")

# Normalverteilung ueberlagern
curve(dnorm(x, mean(r), sd(r)),
      add=TRUE, col="red",
      lwd=2)
```



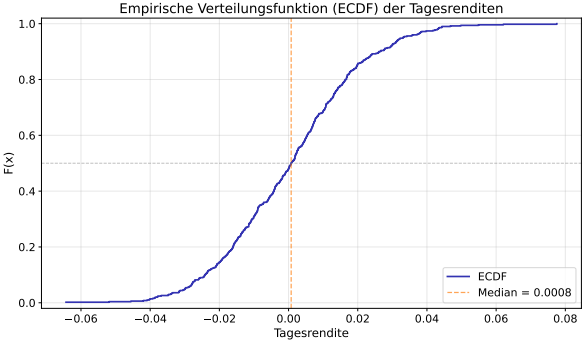
`freq=FALSE` **normiert auf Dichte;** `breaks` **steuert die Anzahl der Klassen.**

Empirische Verteilungsfunktion (ECDF)

plot(ecdf()) erzeugt die empirische kumulative Verteilungsfunktion.

```
# ECDF der Renditen
plot(ecdf(r),
  main="ECDF",
  xlab="Rendite",
  ylab="F(x)",
  col="purple",
  lwd=2)

# Median-Linie
abline(h=0.5,
       lty=2, col="gray")
abline(v=median(r),
       lty=2, col="orange")
```



Die ECDF ist eine Treppenfunktion:
$\hat{F}(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(X_i \leq x)$.

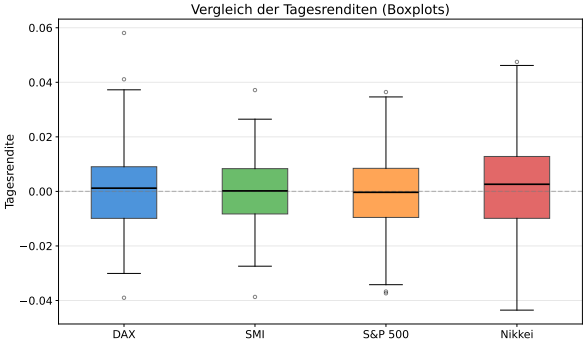
Die ECDF konvergiert gegen die wahre CDF (Satz von Gliwenko-Cantelli).

Boxplot: Vergleich mehrerer Gruppen

boxplot() vergleicht Verteilungen nebeneinander.

```r
set.seed(42)
dax   <- rnorm(250, 0.0003, 0.015)
smi   <- rnorm(250, 0.0002, 0.012)
sp500 <- rnorm(250, 0.0004, 0.014)
nikk  <- rnorm(250, 0.0001, 0.018)

boxplot(dax, smi, sp500, nikk,
  names=c("DAX","SMI",
          "S&P500","Nikkei"),
  main="Renditen-Vergleich",
  ylab="Tagesrendite",
  col=c("steelblue","green3",
        "orange","red"))
abline(h=0, lty=2, col="gray")
```



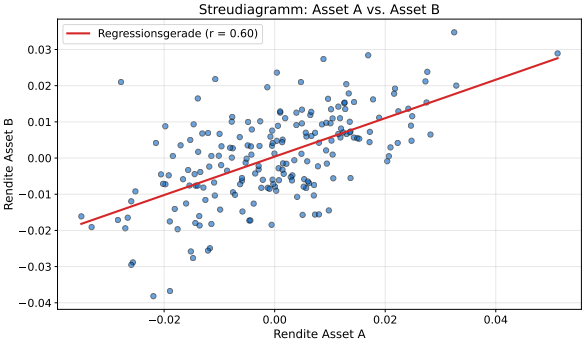
Box = IQR (Q1–Q3); Whiskers = 1.5×IQR; Punkte = Ausreisser.

Streudiagramm und Regressionsgerade

plot(x, y) zeigt den Zusammenhang zweier Variablen.

```r
set.seed(42)
library(MASS)  # fuer mvrnorm
Sigma <- matrix(
  c(0.015^2, 0.6*0.015*0.013,
    0.6*0.015*0.013, 0.013^2),
  nrow=2)
xy <- mvrnorm(200,
  mu=c(0.0003, 0.0002),
  Sigma=Sigma)

plot(xy[,1], xy[,2],
  pch=16, col="steelblue",
  xlab="Rendite A",
  ylab="Rendite B",
  main="Streudiagramm")
abline(lm(xy[,2] ~ xy[,1]),
       col="red", lwd=2)
```



`cor(x, y)` **berechnet den Korrelationskoeffizienten;** `lm()` **die lineare Regression.**

Das d/p/q/r-Schema

R verwendet für **jede** Verteilung vier Funktionen mit einheitlichem Präfix:

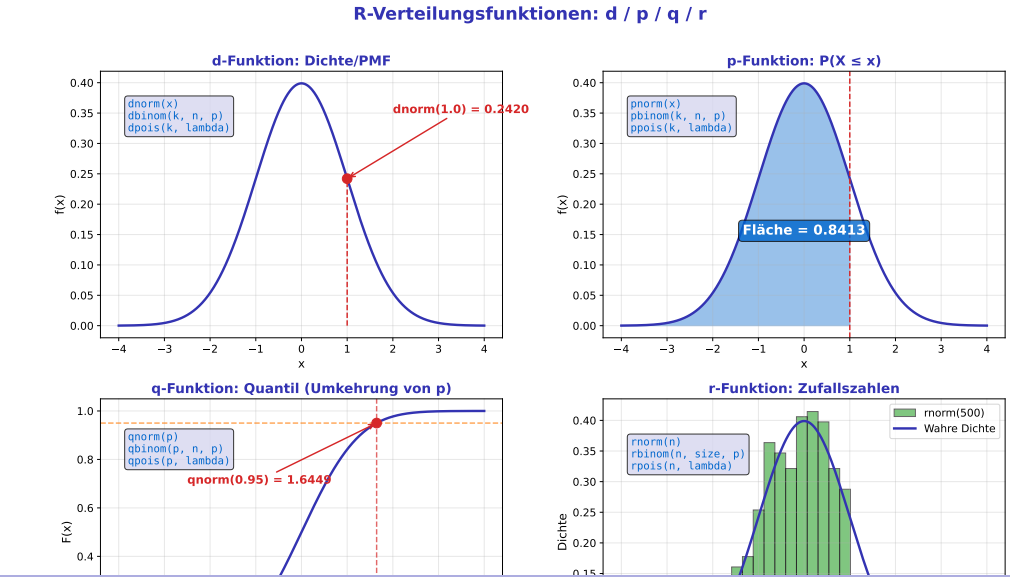
Präfix	Bedeutung	Eingabe	Ausgabe
`d`	**d**ensity / PMF	x oder k	$f(x)$ bzw. $P(X = k)$
`p`	**p**robability (CDF)	x oder k	$P(X \leq x)$
`q`	**q**uantile (Inverse CDF)	Wahrscheinlichkeit p	x mit $P(X \leq x) = p$
`r`	**r**andom	Anzahl n	n Zufallszahlen

Beispiele:

Verteilung	`d`	`p`	`q`	`r`
Binomial	`dbinom`	`pbinom`	`qbinom`	`rbinom`
Poisson	`dpois`	`ppois`	`qpois`	`rpois`
Normal	`dnorm`	`pnorm`	`qnorm`	`rnorm`
Uniform	`dunif`	`punif`	`qunif`	`runif`

Schema merken: dichte, kumulative Probabilität, Quantil, Random.

Das d/p/q/r-Schema – Grafische Übersicht



Binomialverteilung: dbinom und pbinom

$X \sim \text{Bin}(n, p)$: Anzahl Erfolge in n unabhängigen Versuchen mit Erfolgswahrscheinlichkeit p.

Einzelwahrscheinlichkeit (PMF):

```
# P(X = 3) bei n=10, p=0.3
dbinom(3, size=10, prob=0.3)
# Ergebnis: 0.2668

# Alle P(X = k) fuer k = 0..10
dbinom(0:10, size=10, prob=0.3)
```

Kumulative Wahrscheinlichkeit (CDF):

```
# P(X <= 3) bei n=10, p=0.3
pbinom(3, size=10, prob=0.3)
# Ergebnis: 0.6496
```

Obere Wahrscheinlichkeit:

```
# P(X > 3) = 1 - P(X <= 3)
1 - pbinom(3, 10, 0.3)
# Ergebnis: 0.3504

# Oder direkt:
pbinom(3, 10, 0.3,
       lower.tail=FALSE)
# Ergebnis: 0.3504
```

Merke:

- `dbinom`: $P(X = k)$
- `pbinom`: $P(X \leq k)$
- `lower.tail=FALSE`: $P(X > k)$

Die Formel: $P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$

Binomialverteilung: qbinom und rbinom

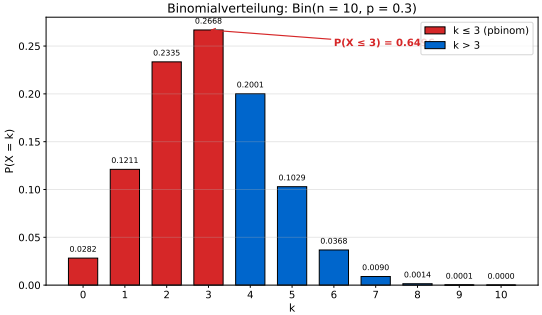
Quantil (Inverse CDF):

```
# Kleinstes k mit P(X <= k) >= 0.95
qbinom(0.95, size=10, prob=0.3)
# Ergebnis: 5

# Das heisst: P(X <= 5) >= 0.95
pbinom(5, 10, 0.3)
# Ergebnis: 0.9527
```

Zufallszahlen:

```
# 1000 Zufallswerte aus Bin(10, 0.3)
set.seed(42)
x <- rbinom(1000, size=10, prob=0.3)
mean(x)    # ungefaehr 3.0
table(x)   # Haeufigkeitstabelle
```



Rot: Werte $k \leq 3$ mit kumulativer Wahrscheinlichkeit $P(X \leq 3) = 0.6496$.

`set.seed()` **macht Zufallszahlen reproduzierbar.**

Poissonverteilung in R

$X \sim \text{Poi}(\lambda)$: Anzahl Ereignisse in einem festen Intervall.

Beispiel: Im Mittel $\lambda = 3$ Schadenfälle pro Monat.

```r
# P(X = 2) bei lambda = 3
dpois(2, lambda=3)
# Ergebnis: 0.2240

# P(X <= 2) bei lambda = 3
ppois(2, lambda=3)
# Ergebnis: 0.4232

# P(X > 5)
ppois(5, 3, lower.tail=FALSE)
# Ergebnis: 0.0839
```

Quantil und Zufallszahlen:

```r
# 95%-Quantil
qpois(0.95, lambda=3)
# Ergebnis: 6

# 500 Zufallswerte
set.seed(42)
x <- rpois(500, lambda=3)
mean(x)    # ungefaehr 3.0
var(x)     # ungefaehr 3.0
```

Eigenschaft der Poissonverteilung:

$$E[X] = \text{Var}(X) = \lambda$$

Die Formel: $P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$ **für** $k = 0, 1, 2, \ldots$

Rechenbeispiel: Diskrete Verteilungen

Aufgabe: Ein Finanzberater kontaktiert 10 Kunden. Jeder Kunde kauft mit Wahrscheinlichkeit $p = 0.3$ ein Produkt.

(a) Genau 4 Kunden kaufen:

```
dbinom(4, size=10, prob=0.3)
# 0.2001
```

(b) Höchstens 3 kaufen:

```
pbinom(3, size=10, prob=0.3)
# 0.6496
```

(c) Mindestens 5 kaufen:

```
1 - pbinom(4, size=10, prob=0.3)
# 0.1503
```

(d) Zwischen 2 und 5 (inklusive):

```
pbinom(5, 10, 0.3) - pbinom(1, 10, 0.3)
# 0.8442
```

(e) Erwartungswert und Varianz:

```
n <- 10; p <- 0.3
n * p           # E(X) = 3
n * p * (1-p)   # Var(X) = 2.1
```

Kontrolle per Simulation:

```
x <- rbinom(100000, 10, 0.3)
mean(x <= 3)    # ~0.6496
```

Tipp: `lower.tail=FALSE` **erspart die Berechnung** $1 - P(X \leq k)$.

Gleichverteilung (Uniform)

$X \sim U(a, b)$: Gleichmässig verteilt auf dem Intervall $[a, b]$.

```r
# Dichte bei x = 0.5, U(0,1)
dunif(0.5, min=0, max=1)
# Ergebnis: 1

# P(X <= 0.7) bei U(0,1)
punif(0.7, min=0, max=1)
# Ergebnis: 0.7

# 30%-Quantil von U(0,1)
qunif(0.3, min=0, max=1)
# Ergebnis: 0.3
```

```r
# 1000 Zufallszahlen aus U(0,1)
set.seed(42)
u <- runif(1000)
hist(u, breaks=20,
     main="Gleichverteilung")

# U(10, 50): z.B. Wartezeit
punif(25, min=10, max=50)
# Ergebnis: 0.375
```

Formeln: $f(x) = \frac{1}{b-a}$, $F(x) = \frac{x-a}{b-a}$

`runif(n)` **ist die Basis vieler Simulationsmethoden.**

Normalverteilung: dnorm und pnorm

$X \sim N(\mu, \sigma^2)$: Die wichtigste stetige Verteilung.

Dichte und Wahrscheinlichkeiten:

```
# Dichte bei x=0 (Standardnormal)
dnorm(0)            # 0.3989

# P(X <= 1.96)
pnorm(1.96)         # 0.9750

# P(X <= -1.96)
pnorm(-1.96)        # 0.0250

# P(-1.96 <= X <= 1.96)
pnorm(1.96) - pnorm(-1.96)
# 0.9500
```

Mit Parametern μ und σ:

```
# N(100, 15^2): IQ-Verteilung
# P(IQ <= 130)
pnorm(130, mean=100, sd=15)
# 0.9772

# P(IQ > 130)
1 - pnorm(130, mean=100, sd=15)
# 0.0228
```

Wichtige Werte:

- $P(|X| \leq 1\sigma) \approx 0.6827$
- $P(|X| \leq 2\sigma) \approx 0.9545$
- $P(|X| \leq 3\sigma) \approx 0.9973$

Standardnormal: `pnorm(x)` **ohne weitere Parameter entspricht** $\mu = 0$, $\sigma = 1$.

Normalverteilung: qnorm und rnorm

Quantile:

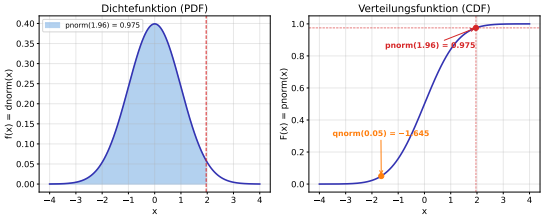
```r
# 5%-Quantil (Standardnormal)
qnorm(0.05)       # -1.6449

# 97.5%-Quantil
qnorm(0.975)      # 1.9600

# 1%-Quantil
qnorm(0.01)       # -2.3263
```

Zufallszahlen:

```r
set.seed(42)
x <- rnorm(10000, mean=0, sd=1)
mean(x)    # ~ 0
sd(x)      # ~ 1
```



Links: PDF mit $P(X \leq 1.96) = 0.975$ (Fläche).
Rechts: CDF mit `qnorm(0.05)` $= -1.645$.

`qnorm(p)` **ist die Umkehrung von** `pnorm(x)`**: gegeben** p**, finde** x**.**

Anwendung: Value at Risk (VaR) mit R

Der **Value at Risk** (VaR) ist das Quantil der Verlustverteilung.

Annahme: Tagesrenditen $\sim N(\mu, \sigma^2)$ mit $\mu = 0.0005$, $\sigma = 0.02$.

```
mu <- 0.0005
sigma <- 0.02

# VaR auf 5%-Niveau
VaR_5 <- qnorm(0.05,
               mean=mu, sd=sigma)
VaR_5
# Ergebnis: -0.03240

# Interpretation: Mit 95% Sicherheit
# ist der Tagesverlust nicht groesser
# als 3.24%
```

```
# VaR auf 1%-Niveau
VaR_1 <- qnorm(0.01,
               mean=mu, sd=sigma)
VaR_1  # -0.04553
```

Formel:

$$\text{VaR}_\alpha = \mu + z_\alpha \cdot \sigma$$

wobei $z_\alpha = \texttt{qnorm}(\alpha)$.

Für ein Portfolio von 1 Mio. EUR:

```
Portfolio <- 1000000
Verlust <- abs(VaR_5) * Portfolio
Verlust  # 32'398 EUR
```

Tipp: `qnorm()` liefert direkt das Quantil – keine manuelle z-Tabelle nötig!

VaR ist ein Standardmass im Risikomanagement (Basel-Regulierung).

Rechenbeispiel: Stetige Verteilungen

Aufgabe: Aktienrenditen seien normalverteilt mit $\mu = 0.08$ (jährlich), $\sigma = 0.20$.

(a) P(Rendite < 0):

```
pnorm(0, mean=0.08, sd=0.20)
# 0.3446
```

(b) P(Rendite > 0.30):

```
1 - pnorm(0.30, mean=0.08, sd=0.20)
# 0.1357
```

(c) 95%-Quantil:

```
qnorm(0.95, mean=0.08, sd=0.20)
# 0.4090
```

(d) Symmetrisches 90%-Intervall:

```
qnorm(c(0.05, 0.95),
      mean=0.08, sd=0.20)
# -0.2490  0.4090
```

(e) Simulation:

```
set.seed(42)
r <- rnorm(100000, 0.08, 0.20)
mean(r < 0)       # ~0.345
quantile(r, 0.95) # ~0.409
```

Simulation bestätigt die analytischen Ergebnisse – ein wichtiges Validierungsinstrument.

curve(): Stetige Verteilungen zeichnen

curve() zeichnet eine mathematische Funktion über ein Intervall.

Standardnormalverteilung:

```
# PDF zeichnen
curve(dnorm(x),
      from=-4, to=4,
      main="N(0,1) Dichte",
      ylab="f(x)",
      col="purple", lwd=2)
```

CDF zeichnen:

```
curve(pnorm(x),
      from=-4, to=4,
      main="N(0,1) CDF",
      ylab="F(x)",
      col="purple", lwd=2)
abline(h=c(0.025, 0.975),
       lty=2, col="red")
```

Mehrere Verteilungen:

```
curve(dnorm(x, 0, 1), from=-6, to=6,
      col="blue", lwd=2,
      main="Vergleich", ylab="f(x)")
curve(dnorm(x, 0, 2), add=TRUE,
      col="red", lwd=2)
curve(dnorm(x, 2, 1), add=TRUE,
      col="green", lwd=2)
legend("topright",
  c("N(0,1)","N(0,4)","N(2,1)"),
  col=c("blue","red","green"),
  lwd=2)
```

Wichtig:

- `add=TRUE`: Kurve zum bestehenden Plot hinzufügen
- `from`, `to`: x-Achsen-Bereich
- `n`: Anzahl Auswertungspunkte (Standard: 101)

`curve()` **ist die schnellste Art, Verteilungen in R zu visualisieren.**

Diskrete Verteilungen zeichnen

Für diskrete Verteilungen nutzt man **barplot()** mit den d-Funktionen.

Binomialverteilung:

```
k <- 0:10
pmf <- dbinom(k, size=10, prob=0.3)

barplot(pmf, names.arg=k,
  main="Bin(10, 0.3)",
  xlab="k", ylab="P(X=k)",
  col="steelblue",
  border="black")
```

Poissonverteilung:

```
k <- 0:15
barplot(dpois(k, lambda=5),
  names.arg=k,
  main="Poi(5)",
  col="orange")
```

Vergleich: Binomial vs. Poisson

Wenn n gross und p klein ist, kann die Poisson-Verteilung die Binomialverteilung approximieren ($\lambda = np$).

```
n <- 100; p <- 0.05
k <- 0:15
binom_pmf <- dbinom(k, n, p)
pois_pmf  <- dpois(k, n*p)

# Nebeneinander plotten
M <- rbind(binom_pmf, pois_pmf)
barplot(M, beside=TRUE,
  names.arg=k,
  col=c("steelblue","orange"),
  main="Bin(100,0.05) vs Poi(5)")
legend("topright",
  c("Binomial","Poisson"),
  fill=c("steelblue","orange"))
```

Faustregel: Poisson-Approximation brauchbar, wenn $n \geq 30$ und $p \leq 0.1$.

Flächen schattieren mit polygon()

Die Funktion **polygon()** schattiert Bereiche unter einer Kurve.

```
# Normalverteilung mit Schattierung
curve(dnorm(x), from=-4, to=4,
      main="P(-1.96 <= X <= 1.96)",
      ylab="f(x)", col="purple", lwd=2)

# Schattierte Flaeche
x_shade <- seq(-1.96, 1.96,
               length.out=200)
y_shade <- dnorm(x_shade)
polygon(c(-1.96, x_shade, 1.96),
        c(0, y_shade, 0),
        col=rgb(0, 0.4, 0.8, 0.3),
        border=NA)

# Beschriftung
text(0, 0.15,
  "95%", cex=1.5, col="blue")
abline(v=c(-1.96, 1.96),
       lty=2, col="red")
```

Anwendung:

- Konfidenzintervalle visualisieren
- p-Werte grafisch darstellen
- Ablehnungsbereiche von Tests zeigen

polygon()-Syntax:

- Erster Vektor: x-Koordinaten
- Zweiter Vektor: y-Koordinaten
- Der Bereich wird geschlossen und gefüllt
- `col=rgb(r,g,b,alpha)` für Transparenz

Technik: Polygon-Punkte beginnen und enden auf der x-Achse ($y = 0$).

QQ-Plot und Normalitätsprüfung

Der **QQ-Plot** vergleicht Stichproben-Quantile mit theoretischen Quantilen.

```r
# Normalverteilte Daten
set.seed(42)
x_norm <- rnorm(200)
qqnorm(x_norm,
  main="QQ-Plot (normal)")
qqline(x_norm, col="red", lwd=2)

# Schiefe Daten (Exponential)
x_exp <- rexp(200, rate=1)
qqnorm(x_exp,
  main="QQ-Plot (exponential)")
qqline(x_exp, col="red", lwd=2)
```

Interpretation:

- Punkte auf der Linie $\Rightarrow$ normalverteilt
- Systematische Abweichungen $\Rightarrow$ nicht normalverteilt
- S-Form $\Rightarrow$ schwere Ränder (fat tails)
- Konkav/konvex $\Rightarrow$ Schiefe

Shapiro-Wilk-Test:

```r
shapiro.test(x_norm)
# p > 0.05: kein Widerspruch
# zur Normalverteilung

shapiro.test(x_exp)
# p < 0.05: nicht normalverteilt
```

QQ-Plots sind oft informativer als formale Tests, da sie die Art der Abweichung zeigen.

Komplettes Beispiel: Verteilungsanalyse

Workflow: Daten → Visualisierung → Verteilungsanpassung → Prüfung.

```r
set.seed(42)
renditen <- rnorm(500, mean=0.0005, sd=0.02)    # Simulierte Tagesrenditen

# 1. Ueberblick
par(mfrow=c(2,2))                                  # 2x2 Layout

# 2. Histogramm mit Normalverteilung
hist(renditen, breaks=30, freq=FALSE, main="Histogramm", col="lightblue")
curve(dnorm(x, mean(renditen), sd(renditen)), add=TRUE, col="red", lwd=2)

# 3. ECDF vs. theoretische CDF
plot(ecdf(renditen), main="ECDF vs. CDF")
curve(pnorm(x, mean(renditen), sd(renditen)), add=TRUE, col="red", lwd=2)

# 4. QQ-Plot
qqnorm(renditen, main="QQ-Plot"); qqline(renditen, col="red", lwd=2)

# 5. Boxplot
boxplot(renditen, main="Boxplot", col="lightblue", horizontal=TRUE)

par(mfrow=c(1,1))                                  # Layout zuruecksetzen
```

`par(mfrow=c(r,c))` **erzeugt ein Raster von** $r \times c$ **Plots.**

Zusammenfassung: d/p/q/r-Funktionen

Verteilung	`d...()`	`p...()`	`q...()`	`r...()`
Binomial	`dbinom(k,n,p)`	`pbinom(k,n,p)`	`qbinom(q,n,p)`	`rbinom(m,n,p)`
Poisson	`dpois(k,`λ`)`	`ppois(k,`λ`)`	`qpois(q,`λ`)`	`rpois(m,`λ`)`
Normal	`dnorm(x,`μ`,`σ`)`	`pnorm(x,`μ`,`σ`)`	`qnorm(q,`μ`,`σ`)`	`rnorm(m,`μ`,`σ`)`
Uniform	`dunif(x,a,b)`	`punif(x,a,b)`	`qunif(q,a,b)`	`runif(m,a,b)`

Wichtige Ergebnisse zum Merken:

- `pnorm(1.96)` $= 0.975$
- `pnorm(-1.96)` $= 0.025$
- `qnorm(0.05)` $= -1.6449$
- `qnorm(0.975)` $= 1.96$
- `pbinom(3, 10, 0.3)` $= 0.6496$
- `ppois(2, 3)` $= 0.4232$

Diese Tabelle ist Ihre Referenz für Klausur und Praxis.

Zusammenfassung: Diagrammtypen in R

Diagramm	R-Funktion	Verwendung
Barplot	`barplot()`	Kategorische Daten, PMF diskreter Verteilungen
Histogramm	`hist()`	Verteilung stetiger Daten
ECDF	`plot(ecdf())`	Empirische Verteilungsfunktion
Boxplot	`boxplot()`	Vergleich mehrerer Gruppen
Streudiagramm	`plot(x, y)`	Zusammenhang zweier Variablen
Funktionskurve	`curve()`	Theoretische Verteilungen
QQ-Plot	`qqnorm()`	Normalitätsprüfung

Tipps:

- `par(mfrow=c(r,c))` für mehrere Plots nebeneinander
- `polygon()` zum Schattieren von Flächen
- `legend()` für Legenden
- `abline()` für Referenzlinien (horizontal, vertikal, Regressionsgerade)

R-Grafiken können mit `pdf()`, `png()` etc. exportiert werden.