WERKZEUGE DER MUSTERERKENNUNG UND DES MASCHINELLEN LERNENS

Vorlesung im Sommersemester 2017

Prof. E.G. Schukat-Talamazzini

Stand: 6. März 2017

Teil IV

Modellieren in R

R-Funktionen für Wahrscheinlichkeitsverteilungen

'R' bietet systematisch **vier** Funktionen je Verteilungsfunktion an

Verteilungsdichtefunktion `dname (x,...)`

Berechnet die Dichtewerte $f(x_i)$ an den Stellen `x[i]`.
Verteilungsparameter werden als (benannte) Argumente übergeben.

Kumulative Verteilungsfunktion `pname (q,...)`

Berechnet Wahrscheinlichkeiten $\mathrm{P}(\mathbb{X} \leq q_i)$ an den Stellen `q[i]`.

Quantile `qname (p,...)`

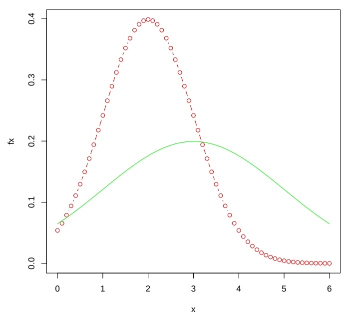
Berechnet die Werte $q_i \in \mathbb{R}$ mit $\mathrm{P}(\mathbb{X} \leq q_i) =$ `p[i]`.

Zufallswerte `rname (n,...)`

Würfelt `n` $\in \mathbb{N}$ viele Zufallswerte unter der spezifizierten Verteilung aus.

Dichtewerte einer Verteilung

`dname(x, «param1» ..... «paramK», log=FALSE)`



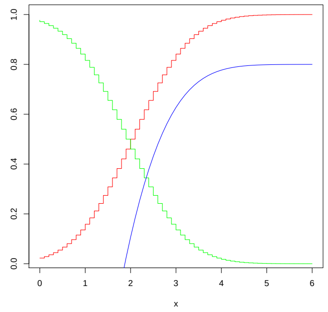

```
x <- seq (0,6,by=0.1)
fx <- dnorm (x, mean=2, sd=1, log=F)
plot (x, fx, col="red", type="b")

gx <- dnorm (x, mean=3, sd=2, log=F)
lines (x, gx, col="green", type="l")
```

Bemerkung

Die Einstellung `log=TRUE` bewirkt die Logarithmierung der Dichtewerte zur Basis e.

Kumulative Verteilungswahrscheinlichkeiten einer Dichte

`pname(q, «param1» ..... «paramK», lower.tail=TRUE, log.p=FALSE)`




```
x <- seq (0,6,by=0.1)
Fx <- pnorm (q=x, mean=2, sd=1)
plot (x, Fx, col="red", type="s")

Gx <- pnorm (q=x, mean=2, sd=1, low=F)
lines (x, Gx, col="green", type="S")

logFx <- pnorm (q=x, mean=2, sd=1, log=T)
lines (x, 0.8+logFx, col="blue", type="l")
```

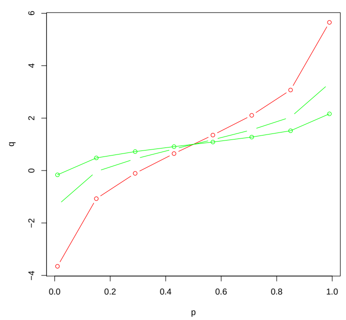
Bemerkung

Die Einstellung `log.p=TRUE` bewirkt die Logarithmierung der Wahrscheinlichkeitswerte zur Basis e.

Mit `lower.tail=FALSE` werden die Werte $\mathrm{P}(\mathbb{X} > q_i)$ statt $\mathrm{P}(\mathbb{X} \leq q_i)$ berechnet.

Quantilwerte einer Verteilung

`qname(p, «param1» ..... «paramK», lower.tail=TRUE, log.p=FALSE)`



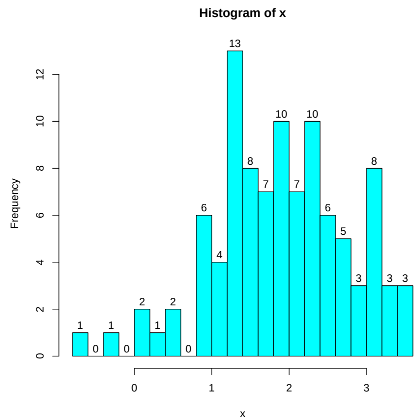

```
p <- seq (0.01,0.99,length=8)
q <- qnorm (p=p, mean=1, sd=2)
plot (p, q, col="red", type="b")

q <- qnorm (p=p, mean=1, sd=1)
lines (p, q, col="green", type="c")

q <- qnorm (p=p, mean=1, sd=0.5)
lines (p, q, col="green", type="o")
```

Bemerkung

Die Einstellung `log.p=TRUE` bewirkt die Interpretation der vorgegebenen Wahrscheinlichkeitswerte als Logarithmen zur Basis e.

1D Sampling — Auswürfeln von Zufallswerten

`rname(n, «param1» ..... «paramK»)`




```
x <- rnorm (100, mean=2, sd=1)

hist (x, breaks=20,
        col="cyan",
        labels=T, plot=T)
```

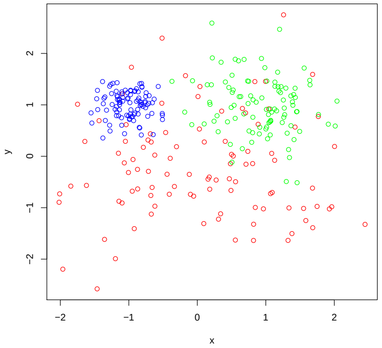
Bemerkung

Mit dem Kommando `set.seed (seed, kind)` kann der **Zufallszahlengenerator** in einen definierten Zustand `seed` $\in \mathbb{N}$ versetzt werden.

`kind` $\in \left\{ \begin{array}{c} \texttt{'Wichmann-Hill'} \\ \texttt{'Marsaglia-Multicarry'} \\ \texttt{'Super-Duper'} \\ \texttt{'Mersenne-Twister'} \\ \texttt{'Knuth-TAOCP-2002'} \end{array} \right\}$

2D Sampling — Auswürfeln von Zufallswerten

`rname(n, «param1» ..... «paramK»)`




```
x <- rnorm (100, mean=0, sd=1)
y <- rnorm (100, mean=0, sd=1)
plot (x, y, col="red", type="p")

x <- rnorm (100, mean=1, sd=0.5)
y <- rnorm (100, mean=1, sd=0.5)
lines (x, y, col="green", type="p")

x <- rnorm (100, mean=-1, sd=0.25)
y <- rnorm (100, mean=1, sd=0.25)
lines (x, y, col="blue", type="p")
```

Verteilungen Statistik reelle WV positive WV [0,1]-WV diskrete WV ?formula N/L-Modelle Optimierung

Sampling — Auswürfeln in endlichen W-Räumen

`sample (x, size, replace=FALSE, prob=NULL)`

- Ziehen ohne Zurücklegen
  - `sample (x=0:9, size=5)` 3 7 1 5 8
  - `sample (x=letters, size=5)` 'm' 'y' 'o' 'f' 'b'
  - `sample (x=49, size=6)` 14 13 30 23 25 45
- Zufallspermutation
  - `sample (x=0:9)` 3 1 4 5 2 9 6 0 7 8
  - `sample (x=6)` 1 6 4 2 5 3
- Ziehen mit Zurücklegen
  - `sample (x=c(T,F), size=5, rep=T)` TRUE FALSE FALSE TRUE FALSE
  - `sample (x=c('C','G','A','T'), size=18, rep=TRUE)`
    'T' 'T' 'G' 'C' 'G' 'A' 'G' 'T' 'C' 'G' 'C' 'C' 'T' 'T' 'A' 'T' 'G' 'C'
- Vorgabe der Proportionen
  - `sample (x=2, size=9, rep=T, prob=c(3,1))` 1 1 2 1 1 1 1 2 1

Verteilungen Statistik reelle WV positive WV [0,1]-WV diskrete WV ?formula N/L-Modelle Optimierung

Arithmetische und getrimmte Mittelwerte

`mean(x, trim=0, na.rm=FALSE, ...)`

- Arithmetisches Mittel einer Zahlmenge
  - `mean (1:100)` [1] 50.5
  - `mean (rnorm (1000, mean=17))` [1] 17.01375
- Mittelwerte der Attribute eines Datensatzes
  `data (iris); mean (iris)`

```
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
5.843333     3.057333    3.758000     1.199333    NA
```

- Mittelwert einer Zahlenmatrix
  - `mean (as.matrix (iris[1:4]))` [1] 3.4645
- Getrimmte Mittelwerte *(Trimm = Anteil der gelöschten unteren/oberen Ausreißer)*
  - `mean (c(1,1,1,4,8), trim=0)` [1] 3
  - `mean (c(1,1,1,4,8), trim=0.2)` [1] 2
  - `mean (c(1,1,1,4,8), trim=0.4)` [1] 1
  - `median (c(1,1,1,4,8))` [1] 1

Varianz und Standardabweichung

`var (x, y=NULL, na.rm=FALSE, use)`

- Varianz eines Datenvektors

```
var (1:5)                                   [1] 2.5
var (rnorm (10**3, mean=7, sd=2))           [1] 4.230118
var (rnorm (10**6, mean=7, sd=2))           [1] 3.997247
```

- Standardabweichung eines Datenvektors

```
sd (rnorm (10**6, mean=7, sd=2))            [1] 2.000575
```

- Kovarianz zweier Datenvektoren

```
x <- rnorm (10**3)
var (x)                                     [1] 0.9386478
var (x, x)                                  [1] 0.9386478
var (x, x+17)                               [1] 0.9386478
var (x, x+x)                                [1] 1.877296
y <- rnorm (10**3)
var (x, y)                                  [1] 0.05117009
```

- (Ko)varianzberechnung nach Pearson

```
sum((x-mean(x))*(y-mean(y)))/(1000-1)       [1] 0.05117009
```


Kovarianz und Korrelation

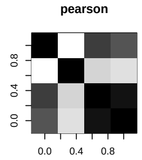
`cov (x, y=NULL, method = c('pearson','kendall','spearman'))`

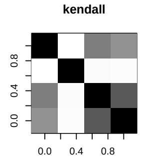
- Kovarianzen für die Spalten einer Matrix

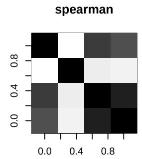
```
cov (iris[1:4])

Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
0.68569351    -0.04243400  1.2743154     0.5162707
-0.04243400   0.18997942   -0.3296564    -0.1216394
1.27431544    -0.32965638  3.1162779     1.2956094
0.51627069    -0.12163937  1.2956094     0.5810063
```

- Korrelationen für die Spalten einer Matrix

```
cor (iris[1:4], method='pearson')

Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
1.0000000     -0.1175698   0.8717538     0.8179411
-0.1175698    1.0000000    -0.4284401    -0.3661259
0.8717538     -0.4284401   1.0000000     0.9628654
0.8179411     -0.3661259   0.9628654     1.0000000
```







Sortierung, Rang und Reihenfolge

- Sortieren von Werten eines Vektors

```
x <- sample (101:109)            108 104 105 107 103 106 102 101 109
sort (x)                         101 102 103 104 105 106 107 108 109
sort (x, decreasing=TRUE)        109 108 107 106 105 104 103 102 101
rev (sort (x))                   109 108 107 106 105 104 103 102 101
```

- Rangnummern ausrechnen

```
rank (x)                         8 4 5 7 3 6 2 1 9
```

- Rangnummern ausrechnen — incl. Doubletten

```
x <- sample (1:9, replace=T)                    6 4 9 1 3 5 9 3 2
rank (x, ties='average')  7.0 5.0 8.5 1.0 3.5 6.0 8.5 3.5 2.0
rank (x, ties='first')                          7 5 8 1 3 6 9 4 2
rank (x, ties='random')                         7 5 8 1 4 6 9 3 2
```

- Sortierindex erstellen

```
print (x)                        6 4 9 1 3 5 9 3 2
order (x)                        4 9 5 8 2 6 1 3 7
x [order (x)]                    1 2 3 3 4 5 6 9 9
```


Quantile — Rangordnungsstatistiken

- **Wertebereichsgrenzen**

```
set.seed (4711)
x <- sample (883:4711, size=101))
range (x)                                          884 4671
```

- **Tukey-Synopse:** 0%,25%,50%,75%,100%-Quantile

```
fivenum (x)                         884 1892 3017 3828 4671
```

- **Quantile** (allgemein)

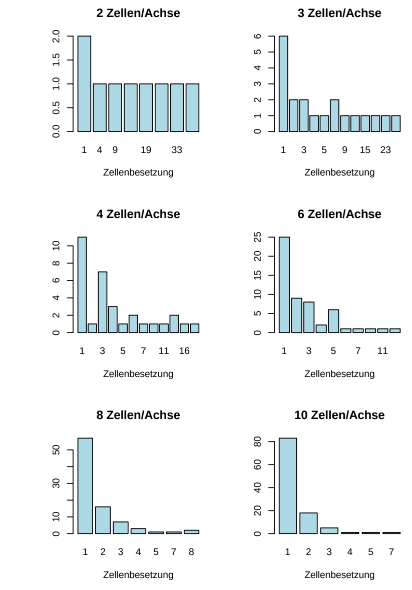
```
quantile (x, probs=0:2/2, type=7)             884 3017 4671
quantile(x,1/2) == median(x)                     50% TRUE :-)
```

- **Univariate Quantisierung**

```
cut (19:11, breaks=3) -> f          (3 Gruppen gleicher Größe)
levels (f)            "(11,13.7]" "(13.7,16.3]" "(16.3,19]"
unclass (f)                               3 3 3 2 2 2 1 1 1
cut (19:11, breaks=3, labels=LETTERS[1:3])  C C C B B B A A A
unclass (cut (19:11, breaks=c(10,15,20)))   2 2 2 2 1 1 1 1 1
```

Beispiel: der Fluch der Dimension

Äquidistante Vergitterung des Merkmalraumes (Hyperkuben)




```
for (m in c(2,3,4,6,8,10)) {
  A <- sapply (
     iris[1:4],
     cut, breaks=m,
     labels=LETTERS[1:m])
  S <- apply (A, MARGIN=1,
     paste, collapse="")
  f <- table (S)
  F <- table (f)
  barplot (F,
     main=paste (m, "Zellen/Achse"),
     xlab="Zellenbesetzung",
     col="lightblue")
  }
```

Verfahrensweise

achsenweise quantisieren	A
Zellencode bilden	S
Zellenbesetzung zählen	f
Größenstatistik berechnen	F

Vergleich zweier Verteilungen

Probability-Probability-Plot *versus* Quantile-Quantile-Plot

Univariate Verteilung

Verteilungsdichte

$$f : \mathbb{R} \to \mathbb{R}_0^+$$

Kumulative Verteil.funktion

$$F : \left\{ \begin{array}{lcl} \mathbb{R} & \to & [0,1] \\ x & \mapsto & F(x) := \int_{-\infty}^{x} f(\xi)\, d\xi \end{array} \right.$$

Quantilfunktion

$$Q : \left\{ \begin{array}{lcl} [0,1] & \to & \mathbb{R} \\ p & \mapsto & Q(p) := F^{-1}(p) \end{array} \right.$$

Verteilungsvergleich:
(f_1, f_2), (F_1, F_2), (Q_1, Q_2) ?

PP-Darstellung

Graph der Punktmenge

$$\langle F_1(x), F_2(x) \rangle_{x \in \mathbb{R}}$$

QQ-Darstellung

Graph der Punktmenge

$$\langle Q_1(p), Q_2(p) \rangle_{x \in [0,1]}$$

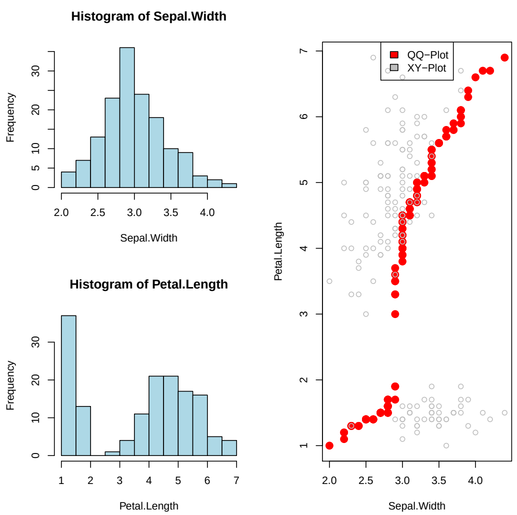
Empirische Formulierung

Datensätze $\left\{ \begin{array}{l} y_1, \ldots, y_m \sim f_1 \\ z_1, \ldots, z_m \sim f_2 \end{array} \right\}$

QQ-Plot	$\langle y_{(j)}, z_{(j)} \rangle$
Scatterplot	$\langle y_j, z_j \rangle$

Beispiel: Verteilungsähnlichkeit vs. Korrelation

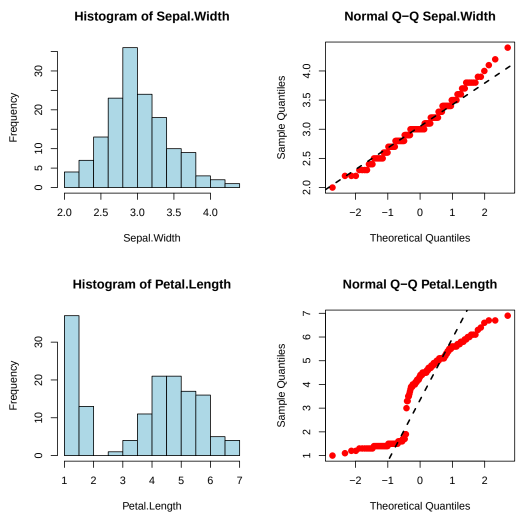
`qqplot (x, y, plot.it=TRUE, xlab=, ylab=, ...)`




```
attach (iris)
layout (matrix (c(1,2,3,3), 2))

hist (Sepal.Width,
   col="lightblue")
hist (Petal.Length,
   col="lightblue")

qqplot (
   Sepal.Width, Petal.Length,
   col="red", pch=19, cex=1.5)
points (
   Sepal.Width, Petal.Length,
   col="gray")
legend ("top",
   legend=c("QQ-Plot","XY-Plot"),
   fill=c("red","gray"))
```


Normalverteilungseigenschaft einer Punktmenge

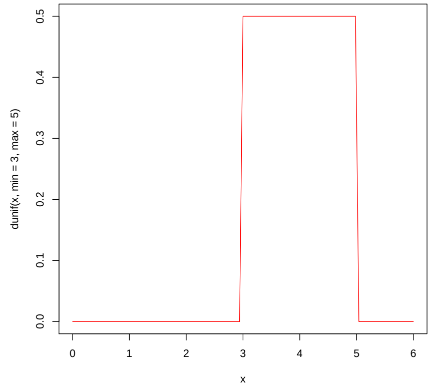
`qqnorm (y, ylim, main=, xlab=, ylab=, plot.it=TRUE, datax=FALSE, ...)`




```
attach (iris)
layout (matrix (c(1,2,3,4), 2))

hist (Sepal.Width,
   col="lightblue")
hist (Petal.Length,
   col="lightblue")

qqnorm (Sepal.Width,
   main="Normal Q-Q Sepal.Width",
   col="red", pch=19, cex=1.2)
qqline (Sepal.Width, lty=2, lwd=2)

qqnorm (Petal.Length,
   main="Normal Q-Q Petal.Length",
   col="red", pch=19, cex=1.2)
qqline (Petal.Length, lty=2, lwd=2)
```

Gleich-, Rechteck- oder uniforme Verteilung

`dunif (x, min=0, max=1)`

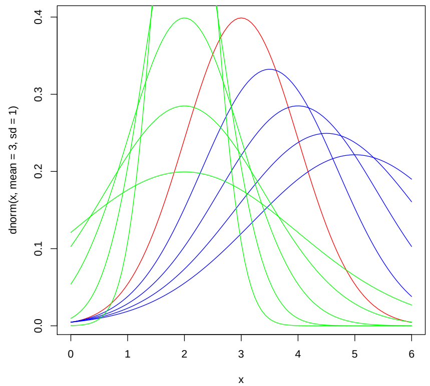


$x \in \mathbb{R}$
a Minimalwert
b Maximalwert
$\mathcal{E}[\mathbb{X}] = {}^{(a+b)}/_{2}$
$\mathrm{Var}[\mathbb{X}] = {}^{(b-a)}/_{12}$

$$f_{a,b}(x) = \frac{1}{b-a}$$

Normalverteilung

`dnorm (x, mean=0, sd=1)`

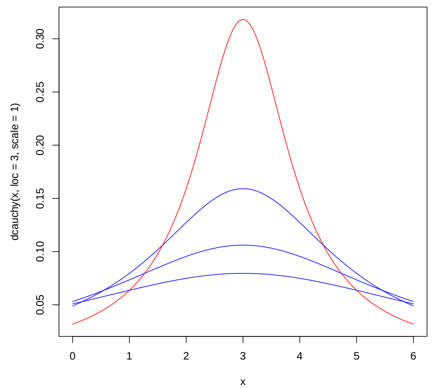


$x \in \mathbb{R}$
μ Erwartungswert
$\sigma > 0$ Standardabweichung
$\mathcal{E}[\mathbb{X}] = \mu$
$\mathrm{Var}[\mathbb{X}] = \sigma^2$

$$f_{\mu,\sigma}(x) = \mathcal{N}(x \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left\{-\frac{1}{2} \cdot \left(\frac{x-\mu}{\sigma}\right)^2\right\}$$

Cauchyverteilung

`dcauchy (x, location=0, scale=1)`



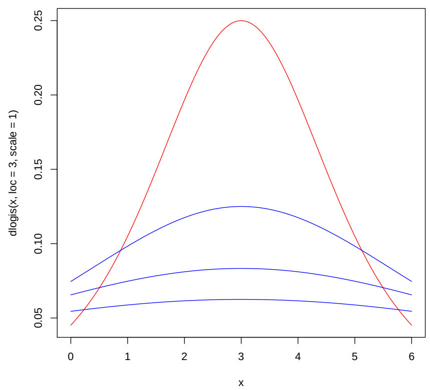
$x \in \mathbb{R}$
μ Lageparameter
s Skalenparameter
$\mathcal{E}[\mathbb{X}] = \mu$
$\mathrm{Var}[\mathbb{X}] = \infty$

$$f_{\mu,s}(x) = \left\{\pi s \cdot \left(1 + \left(\frac{x-\mu}{s}\right)^2\right)\right\}^{-1}$$

„Schwarzeneggerdichte“ (schmaler Kopf, extrem breite Schultern)

Logistische Verteilung

`dlogis (x, location=0, scale=1)`



$x \in \mathbb{R}$

μ Lageparameter

s Skalenparameter

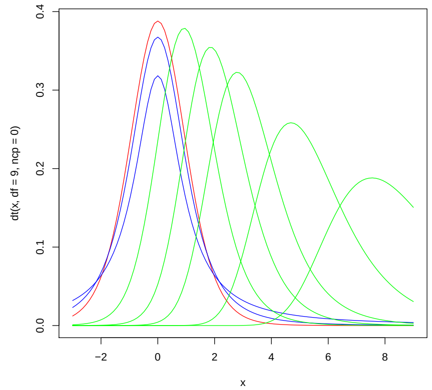
$\mathcal{E}[\mathbb{X}] = \mu$

$\mathrm{Var}[\mathbb{X}] = {}^{\pi^2}\!/_3 \cdot s^2$

$$f_{\mu,s}(x) = \frac{1}{s} \cdot e^{\frac{x-\mu}{s}} \cdot \left\{1 + e^{\frac{x-\mu}{s}}\right\}^{-2} \quad \text{bzw.} \quad F_{\mu,s}(x) = \left\{1 + e^{-\frac{x-\mu}{s}}\right\}^{-1}$$

Students t-Verteilung

`dt (x, df, ncp=0)`



$x \in \mathbb{R}$

n Anzahl Freiheitsgrade

`ncp` Nicht-Zentralität

$\mathcal{E}[\mathbb{X}] = 0$

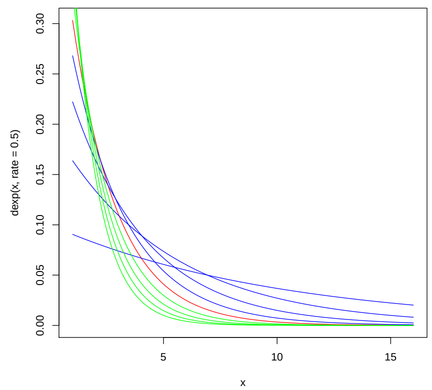
$\mathrm{Var}[\mathbb{X}] = {}^{n}\!/_{(n-2)}$

$$f_n(x) = \frac{\Gamma\left(\frac{n+1}{2}\right)}{\sqrt{n\pi} \cdot \Gamma\left(\frac{n}{2}\right)} \cdot \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}$$

Die Variable $\mathbb{T} = (\hat{\mu} - \mu_0) \cdot \sqrt{n}/\hat{\sigma}$ von n i.i.d. $\mathcal{N}(\mu, \sigma^2)$-verteilten ZVen ist Student $(n-1, t^*)$-verteilt mit $t^* = (\mu - \mu_0) \cdot \sqrt{n}/\sigma$

Exponentialverteilung

`dexp (x, rate=1)`



$x \geq 0$

λ Abklingrate

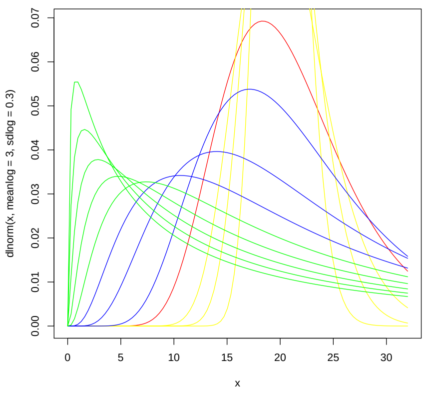
$\mathcal{E}[\mathbb{X}] = {}^{1}\!/_{\lambda}$

$\mathrm{Var}[\mathbb{X}] = {}^{1}\!/_{\lambda^2}$

$$f_\lambda(x) = \lambda \cdot \exp\{-\lambda \cdot x\}$$

Lognormalverteilung

`dlnorm (x, meanlog=0, sdlog=1)`



$x \quad \geq 0$

$\mu \quad$ Mittel (Logskala)

$\sigma^2 \quad$ Varianz (Logskala)

$\mathcal{E}[\mathbb{X}] \; = \exp(\mu + {}^{\sigma^2}\!/_2)$

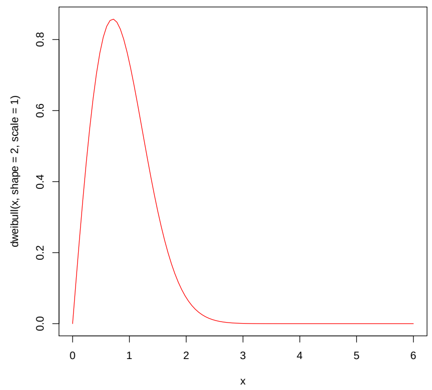
$\mathrm{Var}[\mathbb{X}] \; = \exp(2\mu+\sigma^2)\cdot(\exp(\sigma^2)-1)$

$$f_{\mu,\sigma}(x) \;=\; \frac{\mathcal{N}(\log x \mid \mu, \sigma)}{x} \;=\; \frac{1}{x \cdot \sigma \cdot \sqrt{2\pi}} \cdot \exp\left\{-\frac{(\log x - \mu)^2}{2\sigma^2}\right\}$$

$\mathbb{X}$ ist (μ, σ)-lognormalverteilt genau dann, wenn $\log(\mathbb{X}) \sim \mathcal{N}(\mu, \sigma)$

Weibullverteilung

`dweibull (x, shape, scale=1)`



$x \quad \geq 0$

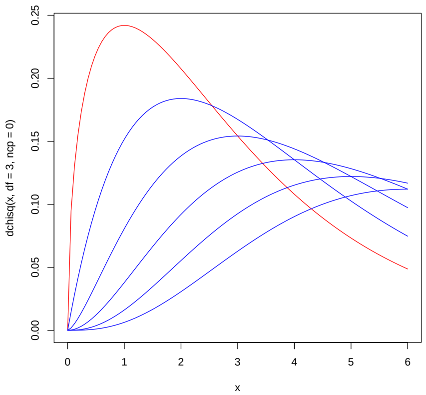
$a \quad$ Formparameter

$s \quad$ Skalenparameter

$$f_{a,s}(x) \;=\; \frac{a}{s} \cdot \left(\frac{x}{s}\right)^{a-1} \cdot \exp\left\{-\left(\frac{x}{s}\right)^{a}\right\}$$

Zentrale χ^2-Verteilung

`dchisq (x, df)`



$x \quad \geq 0$

$n \quad \#$ Freiheitsgrade

$\mathcal{E}[\mathbb{X}] \; = n$

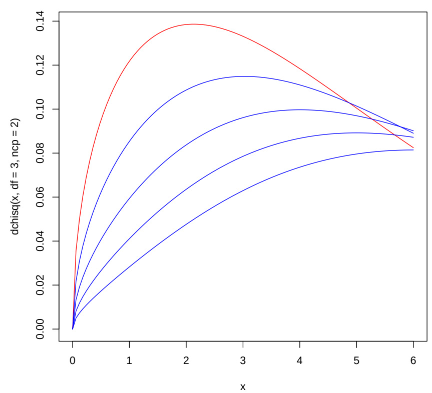
$\mathrm{Var}[\mathbb{X}] \; = 2n$

$$f_n(x) \;=\; \chi^2_n(x) \;=\; \frac{1}{2^{n/2} \cdot \Gamma({}^{n}\!/_2)} \cdot x^{n/2-1} \cdot e^{-x/2}$$

Summe der Quadrate von n vielen $\mathcal{N}(0,1)$-verteilten unabhängigen Zufallsvariablen

Nichtzentrale χ^2-Verteilung

`dchisq (x, df, ncp=0)`



$x \quad \geq 0$

$n \quad \#$ Freiheitsgrade

$\lambda \quad$ Nicht-Zentralität

$\mathcal{E}[\mathbb{X}] \; = n$

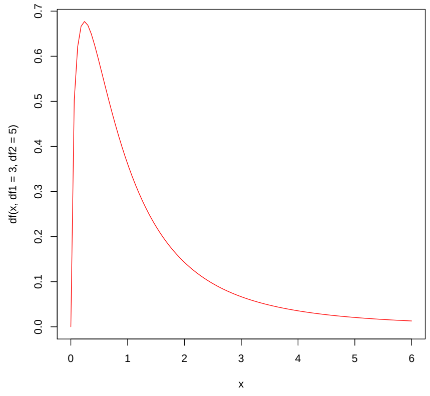
$\mathrm{Var}[\mathbb{X}] \; = 2n$

$$f_{n,\lambda}(x) \;=\; \chi^2_{n,\lambda}(x) \;=\; e^{-\lambda/2} \cdot \sum_{r=0}^{\infty} \frac{(\lambda/2)^r}{r!} \cdot \chi^2_{n+2r}(x)$$

Summe der Quadrate von n vielen $\mathcal{N}(\lambda,1)$-verteilten unabhängigen Zufallsvariablen

F-Verteilung

`df (x, df1, df2)`



$x \geq 0$

n Freiheitsgrade Zähler

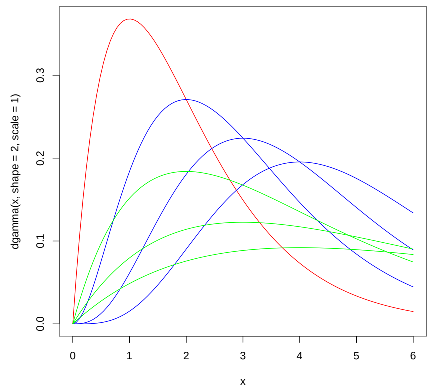
m Freiheitsgrade Nenner

$$f_{n,m}(x) = \frac{\Gamma\left(\frac{n+m}{2}\right)}{\Gamma\left(\frac{n}{2}\right)\cdot\Gamma\left(\frac{m}{2}\right)} \cdot \left(\frac{n}{m}\right)^{n/2} \cdot x^{n/2-1} \cdot \left(1 + \frac{n}{m}x\right)^{-\frac{n+m}{2}}$$

Quotient der mittleren Quadratsummen von n bzw. m unabhängigen $\mathcal{N}(0,1)$-verteilten Zufallsvariablen

Gammaverteilung

`dgamma (x, shape, rate=1, scale=1/rate)`



$x \geq 0$

$a > 0$ Formparameter

$s > 0$ Skalenparameter

$r > 0$ Rate $r = 1/s$

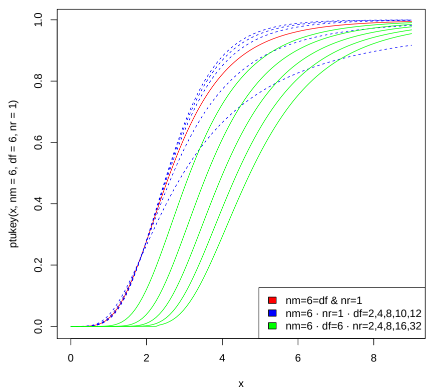
$\mathcal{E}[\mathbb{X}] = a \cdot s$

$\mathrm{Var}[\mathbb{X}] = a \cdot s^2$

$$f_{a,s}(x) = \gamma_{a,s}(x) = \frac{1}{s^a \cdot \Gamma(a)} \cdot x^{a-1} \cdot e^{-x/s}$$

Tukeys studentisierte Spannenverteilung

`ptukey (q, nmeans, df, nranges=1, lower.tail=TRUE, log.p=FALSE)`



$q \geq 0$

$nmeans$ Umfang m der $\mathcal{N}(0,1)$-Probe(n)

df Freiheitsgrade der Studentisierung

$nranges$ Anzahl n gezogener Proben

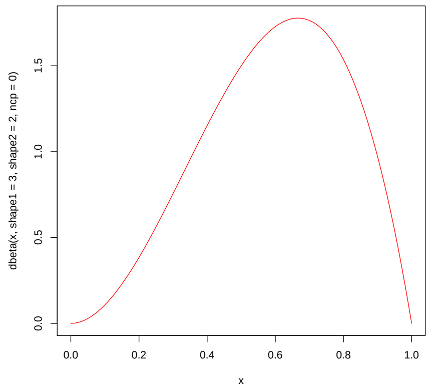
$$F_{m,df,n}(x) \stackrel{\text{def}}{=} \mathrm{P}(R/s \leq x)$$

R ist die maximale Spannweite unter n gezogenen $\mathcal{N}(0,1)$-Proben der Größe m.

$df \cdot s^2$ ist (unabhängig davon) χ^2-verteilt mit df Freiheitsgraden.

Betaverteilung

`dbeta (x, shape1, shape2, ncp=0)`

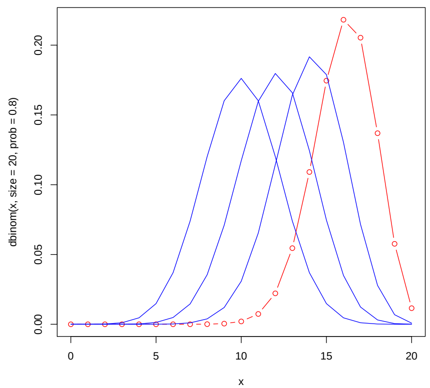


x	$\in [0,1]$
$a > 0$	Formparameter #1
$b > 0$	Formparameter #2
`ncp`	Nicht-Zentralität

$$p_{a,b}(x) \;=\; \beta(x \mid a, b) \;=\; \frac{\Gamma(a+b)}{\Gamma(a)\cdot\Gamma(b)} \cdot x^{a-1} \cdot (1-x)^{b-1}$$

Binomialverteilung

`dbinom (x, size, prob)`



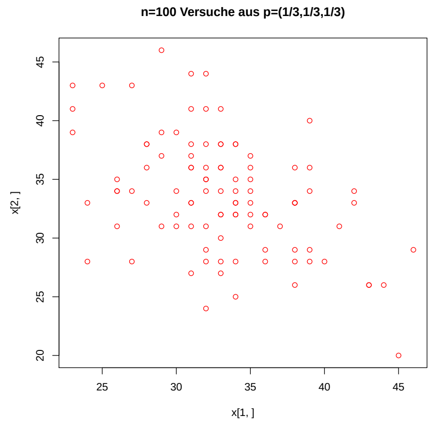
x	$\in \{0,1,2,\ldots,n\}$
n	Bernoulli-Versuche
p	Trefferwahrscheinlichkeit
$\mathcal{E}[\mathbb{X}]$	$= n \cdot p$
$\mathrm{Var}[\mathbb{X}]$	$= n \cdot p \cdot (1-p)$

$$p_{n,p}(x) \;=\; \mathcal{B}(x \mid n, p) \;=\; \binom{n}{x} \cdot p^x \cdot (1-p)^{n-x}$$

Ziehen von n Kugeln **mit** Zurücklegen aus einer $(p, 1-p)$-Urne

Multinomialverteilung

`dmultinom (x, size=sum(x), prob)`



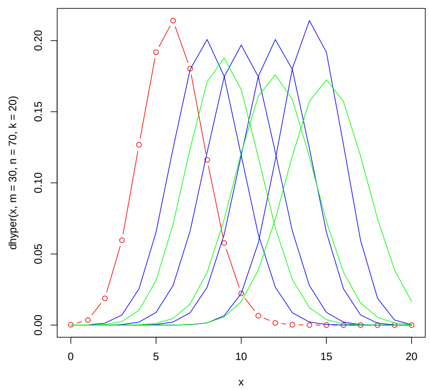
$\boldsymbol{x}$	$\in \mathbb{N}^K$
n	$= \sum_k x_k$ Versuche
$\boldsymbol{p} \geq \boldsymbol{0}$	kanonische Parameter
$\mathcal{E}[\mathbb{X}_k]$	$= n \cdot p_k$
$\mathrm{Cov}[\mathbb{X}_i, \mathbb{X}_j]$	$= -n \cdot p_i p_j$

$$p_{n,\boldsymbol{p}}(\boldsymbol{x}) \;=\; \mathcal{B}(\boldsymbol{x} \mid n, \boldsymbol{p}) \;=\; \binom{n}{\boldsymbol{x}} \cdot \prod_{k=1}^{K} p_k^{x_k} \;=\; \frac{(\sum_k x_k)!}{\prod_k x_k!} \cdot \prod_{k=1}^{K} p_k^{x_k}$$

Ziehen von n Kugeln **mit** Zurücklegen aus einer $(p_1, \ldots, p_K)$-Urne

Hypergeometrische Verteilung

`dhyper (x, m, n, k)`



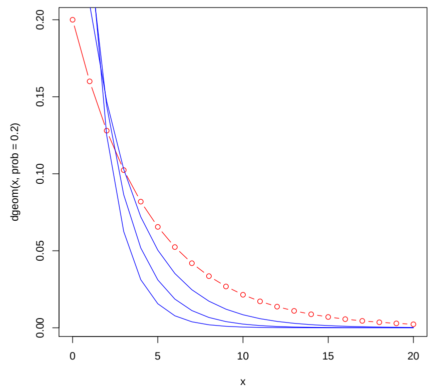
x	$\in \{0,1,2,\ldots,k\}$
m	weiße Kugeln
n	schwarze Kugeln
k	gezogene Kugeln

$$p_{n,m,k}(x) \;=\; \frac{\binom{m}{x}\cdot\binom{n}{k-x}}{\binom{m+n}{k}}$$

Ziehen von k Kugeln **ohne** Zurücklegen aus einer (m,n)-Urne

Geometrische Verteilung

`dgeom (x, prob)`



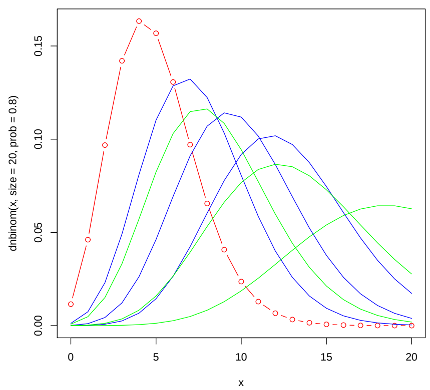
x	$\in \mathbb{N}$
p	Trefferwahrscheinlichkeit
$\mathcal{E}[\mathbb{X}]$	$= {}^{(1-p)}/_{p}$
$\mathrm{Var}[\mathbb{X}]$	$= {}^{(1-p)}/_{p^2}$

$$p_p(x) \;=\; p\cdot(1-p)^x$$

Anzahl x der Fehlversuche bevor der erste Treffer gelandet wurde

Negative Binomialverteilung

`dnbinom (x, size, prob, mu)`



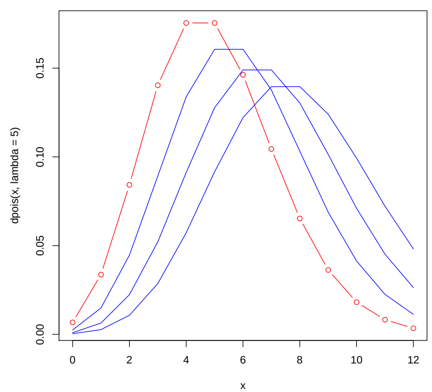
x	$\in \mathbb{N}$
n	Trefferzahl
p	Trefferwahrscheinlichkeit
μ	Erwartungswert (statt p)
$\mathcal{E}[\mathbb{X}]$	$= {}^{n}/_{p} - n$
$\mathrm{Var}[\mathbb{X}]$	$= \mu + {}^{\mu^2}/_{n}$

$$p_{n,p}(x) \;=\; \frac{\Gamma(x+n)}{\Gamma(n)\cdot x!}\cdot p^n\cdot(1-p)^x$$

Anzahl x der Fehlversuche bevor der n-te Treffer gelandet wurde

Poissonverteilung

`dpois (x, lambda)`



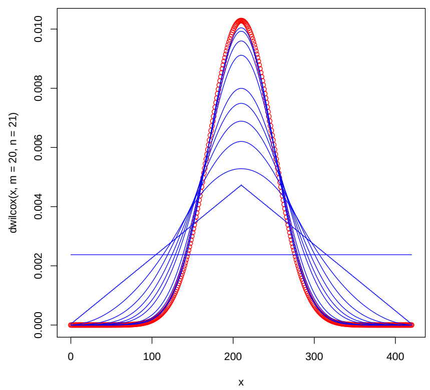
x	$\in \mathbb{N}$
$\lambda > 0$	Formparameter
$\mathcal{E}[\mathbb{X}]$	$= \lambda$
$\mathrm{Var}[\mathbb{X}]$	$= \lambda$

$$p_\lambda(x) \;=\; \lambda^x\cdot\exp\left\{-\frac{\lambda}{x!}\right\}$$

Anzahl x der Treffer eines seltenen $(p\to 0)$ Ereignisses nach $n = {}^{\lambda}/_{p}$ Versuchen

Wilcoxon Rangsummenverteilung

`dwilcox (x, m, n)`



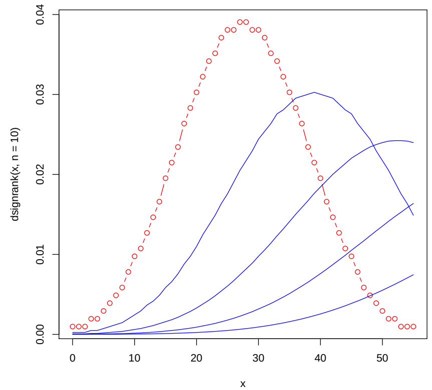
$$\begin{aligned} x &\in \{0, 1, 2, \dots, mn\} \\ m \in \mathbb{N} &\ \text{Probenumfang} \\ n \in \mathbb{N} &\ \text{Probenumfang} \\ \mathcal{E}[\mathbb{X}] &= {}^{mn}\!/_{2} \\ \mathrm{Var}[\mathbb{X}] &= (m+n+1) \cdot {}^{mn}\!/_{12} \end{aligned}$$

$$p_{m,n}(x) \;=\; \mathrm{P}\Big(\sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{I}_{\mathbb{A}_i \geq \mathbb{B}_j} \;=\; x\Big)$$

Für wieviele Paare (a_i, b_j) der i.i.d. Daten $\boldsymbol{a} \in \mathbb{R}^m$, $\boldsymbol{b} \in \mathbb{R}^n$ gilt $a_i \geq b_j$?

Wilcoxon Vorzeichen-Rangsummenverteilung

`dsignrank (x, n)`



$$\begin{aligned} x &\in \{0, 1, 2, \dots, (n+1\}\cdot {}^{n}\!/_{2}) \\ n \in \mathbb{N} &\ \text{Probenumfang} \\ \mathcal{E}[\mathbb{X}] &= (n+1) \cdot {}^{n}\!/_{4} \\ \mathrm{Var}[\mathbb{X}] &= (n+1)(2n+1) \cdot {}^{n}\!/_{24} \end{aligned}$$

$$p_{n}(x) \;=\; \mathrm{P}\Big(\sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{I}_{|\mathbb{A}_i| \geq |\mathbb{A}_j|} \cdot \mathbf{I}_{\mathbb{A}_i > 0} \;=\; x\Big)$$

Wie groß ist die Summe der **Absolutränge** aller positiven Zahlen a_i ?

Daten, Wahrscheinlichkeiten und Vorhersage

Auswürfeln einer Stichprobe · Maximum-Likelihood-Schätzung

Wahrscheinlichkeitsmodell

Multivariate Verteilungsdichte

$$f : \begin{cases} \mathbb{R}^n & \rightarrow \quad \mathbb{R}_0^+ \\ \boldsymbol{x} & \mapsto \quad f(x_1, \dots, x_n) \end{cases}$$

Datensatz

Matrix (Objekte × Merkmale)

$$\boldsymbol{X} \;=\; \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{pmatrix}$$

Modell ➪ Daten

Ziehen (ohne Zurücklegen)

$$f(\boldsymbol{X}) \;=\; \prod_{i=1}^{m} f(x_{i1}, \dots, x_{in})$$

i.i.d. $\hat{=}$ *independ. ident. distrib.*

Daten ➪ Modell

Maximale Datenerzeugungswahrscheinlichkeit

$$\hat{f}_{\mathrm{ML}} \;=\; \underset{f \in \mathcal{M}}{\operatorname{argmax}}\, f(\boldsymbol{X})$$

$\mathcal{M}$ Menge (parametrischer) Verteilungskandidaten

Kettenregel und Vorhersagemodell

Kettenregel der Wahrscheinlichkeitstheorie

Produkt **bedingter** Wahrscheinlichkeiten *(a posteriori)*

$$f(x_1, \ldots, x_n) \;=\; \prod_{j=1}^{n} f(x_j \mid x_1, \ldots, x_{j-1})$$

multivariate Statistik ⇦ Prädiktion: $y \leftarrow x_1, \ldots, x_n$

Regressionsanalyse

Wahrscheinlichster y-Wert

$$\hat{y}(\boldsymbol{x}) \;=\; \mathcal{E}_{f(\mathbb{Y}|\boldsymbol{x})}[\mathbb{Y}]$$

bei gegebenen Quellvariablen

Ausgleichsrechnung (OLS)

min. quadrat. Vorhersagefehler

$$\sum_{i=1}^{m} (h(\boldsymbol{x}_i, \boldsymbol{a}) - y_i)^2 \overset{!}{\to} \text{MIN}$$

Bemerkung

Gleichwertig: **Regression** mit $(\mathbb{Y}, \mathbb{X}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{S})$ und **OLS** mit $h(\boldsymbol{x}, \boldsymbol{a}) = a_0 + \sum_j a_j x_j$

Vorhersagemodelle

„Eierlegende Wollmilchsau“ für Mustererkennung, Data Mining & Künstliche Intelligenz

Interpolation von Funktionsverläufen

Stützstellenwerte $y_i = h(\boldsymbol{x}_i)$ unbekannter funktionaler Abhängigkeiten $h(\cdot)$

Modellierung beobachteter Funktionswerte

Verrauschte Beispiele $y_i \approx h(\boldsymbol{x}_i)$ unbekannter funktionaler Abhängigkeiten $h(\cdot)$

- \+ Glättung des Funktionsverlaufs
- \+ Korrelation und Abhängigkeiten höherer Ordnung
- \+ globales Datenmodell nach Kettenregel
- \+ Vorhersage zukünftiger Werte bei Zeitreihen (Extrapolation)

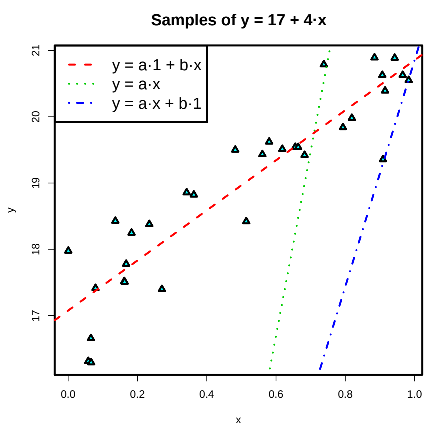
Klassifikation

$x_1, \ldots, x_n \in \mathbb{R}^n$ Mustermerkmale, $y \in \{1 : K\}$ Klassenvariable

Lineares Vorhersagemodell

Linearkombination der Quellvariablen & Versatz (*„intercept“*) in y-Richtung

$$\mathbb{Y}|_{\boldsymbol{x}} \;=\; h(\boldsymbol{x}, \boldsymbol{a}) + \mathbb{E} \;=\; a_0 + \sum_{j=1}^{n} a_j \cdot x_j + \mathbb{E} \qquad \begin{cases} a_0 & h(\boldsymbol{0}, \boldsymbol{a}) \\ a_j & x_j\text{-Steigung} \\ \mathbb{E} & \sim \mathcal{N}(0, \sigma^2) \end{cases}$$



Univariates OLS-Modell

Ausgleichsgerade $\hat{y} = a + b \cdot x$

```
h <- function (x, a=17, b=4) a+b*x
x <- matrix (runif (32), ncol=1)
y <- h(x) + rnorm (length(x), mean=0, sd=0.5)

plot (x, y, pch=24, bg="cyan",
      main="Samples of y = 17 + 4·x")

abline (reg=lm.fit (x, y), col=3, lty=3)
abline (reg=lm.fit (cbind(1,x), y), col=2, lty=2)
abline (reg=lm.fit (cbind(x,1), y), col=4, lty=4)

legend ("topleft", lty=2:4, col=2:4,
      cex=1.5, legend=leg.text)
```


Lineares OLS-Modell in R

```
lm.fit (x, y, offset=NULL, method='qr', tol=1e-7, singular.ok=TRUE, ...)
```

Beispielaufruf

```
lm.fit (cbind (1, as.matrix (iris[-5])), iris$Species=='setosa')
```

⇝ Listenobjekt `lobj` mit Einträgen (unter anderem):

- `lobj$coefficients` `0.120 0.066 0.240 -0.220 -0.057`
  Modellkoeffizienten a_j (je x-Spalte)
- `lobj$fitted.values` `0.979 0.844 0.902 0.826 0.997 1.017 ... 0.008 -0.013`
  Vorhersagewerte $\hat{y}_i = \boldsymbol{a}^\top \boldsymbol{x}_i$ (je x-Zeile)
- `lobj$residuals` `0.021 0.156 0.098 0.174 0.003 -0.017 ... -0.008 0.013`
  Soll-Ist-Differenzen $r_i = y_i - \hat{y}_i$ (je x-Zeile)

Problem

Umständliche Datenmatrixkonstruktion — *Spalten, 1=Versatz, Interaktionen?*
Explizite (unökonomische) Expansion — *Polynomterme!!*
Nichtnumerische Attributskalen — *multiple Binärkodierung von Faktoren*
Wenig intuitive Aufrufsyntax

Modellformeln

Kompakte Notation für Vorhersagemodelle

Syntax für Modellformeln

«formula» ::= «response» ˜ «explaining variables»

- **Zielvariable *(response)***
  sind im Namensraum bekannte Datenvektoren
  `x, 1:12, Sepal.Length, Species`
- **Menge erklärender Variablen *(explaining)***
  ist sprachlicher Ausdruck für eine Kombination von Termen
- **Vereinigungsoperator**
  `Sepal.Length + Petal.Width + Species`
  (die Konstante '1' ist immer implizit dabei)
- **Differenzoperator**
  `Sepal.Length + Petal.Width - 1`

Lineare (Quadratmittel-)Modelle

`lm (formula, data, subset, weights, ...)`

- **Vorhersagemodell für verrauschte Daten**
  ```
  x <- 1:100
  y <- 4*x + 17 + rnorm(length(x))
  ```
  Gesucht: die Koeffizienten des LSE-Modells
- **Berechnung eines affinen Modells**
  `lm.aff <- lm (y ˜ x)`
- **Berechnung eines linearen Modells**
  `lm.lin <- lm (y ˜ x-1)`
- **Berechnung mittels Datensatzvariablen**
  `lm.iris <- lm (Petal.Width ˜ Sepal.Width, data=iris)`
- **Alle außer Zielvariable**
  `lm.iris <- lm (Petal.Width ˜ ., data=iris[-5])`

Modellgetriebene Vorhersage

`predict.lm (object, newdata, ...)`

- **Vorhersage für die Lerndaten des Modells**
  `guess <- predict (object=lm.aff)`
  (polymorpher Aufruf mit `lm`-Objekt)
- **Vorhersage für frische Eingabedaten**
  `guess <- predict (lm.aff, newdata=list(x=1:5))`
  (Datensatz mit passenden Variablennamen!)
- **Vorhersage für Datensätze**
  `guess <- predict (lm.iris, newdata=iris[1:50,-5])`

Interaktionsterme in Modellformeln

Mengenalgebra versus Variablenarithmetik

- **Elementare Termmengen**
  Variablennamen, Intercept '1', Punktoperator '.'
- **Termmengenalgebra**
  Plus '+', Minus '-', runde Klammern
  `(x+y+z)+(x+z)-z` `x, y, 1`
- **Interaktionsterme**
  alle paarweisen Produkte, keine Doubletten, keine Quadrate
  `(x+y+z):(x+z)` `x:z, y:x, y:z, 1`
- **Kumulative Interaktionsterme**
  `(x+y+z)*(x+z)` `x, y, z, x:z, y:x, y:z, 1`
- **Potenzoperator**
  `(x+y+z)^3` `1, x, y, z, x:y, x:z, y:z, x:y:z`

Variablenberechnungen in Modellformeln

Mengenalgebra versus Variablenarithmetik

Variablentyp

Für numerische Variable gilt *Interaktion* ≙ *Produktbildung*.

- **Der Inhibit-Operator**
  verhindert in Formulaausdrücken eine Mengeninterpretation
  `I(x*y)*z` — `1, I(x*y), z, I(x*y):z`
- **Erzeugung von Quadraten und Kuben**
  `I(x^3)+I(y)^3+z^3` — `1, I(x^3), I(y), z`
- **Versehentliche Erzeugung von Doubletten**
  `I(x)*x` — `1, I(x), x, I(x):x`
- **Nicht schutzbedürftige Arithmetik**
  `log(y) ~ log(x)+x` — `... 1, I(log(x)), x`

Modellformeln

Konstruktion, Konversion & sonstige Funktionalitäten

- **Formel versus Zeichenkette**
  `class (y ~ x+z)` — `[1] 'formula'`
  `class ('y ~ x+z')` — `[1] 'character'`
- **Konstruktor**
  Aufruf insbesondere mit Zeichenkettenargument
  `formula ('y ~ x+z')` — `y ~ x+z`
- **Zeichenkettendarstellung**
  `as.character (y ~ x+z)` — `[1] '~' 'y' 'x+z'`
- *Sonstiges:*
  `|`-Operator für bedingte Modelle ⇝ `?coplot`
  `%in%`-Operator für geschachtelte Modelle
  „*specials*“ `offset`, `strata`, `cluster` ⇝ `?terms.formula`

Modellklassen und ihre Standardmethoden

Beispiel: Lineares OLS-Modell `lm()` und die `iris`-Daten

„*Setosa*“-Blüte (50) oder nicht (100) — das ist hier die Frage:

`lm (Species=="setosa" ~ ., data=iris) -> o` `is.factor (Species)` !

- **Vorhersage** (der Klassenzugehörigkeit)
  `set <- predict (o); sum ((set>0.5) == (Species=="setosa"))` — `150`
- **Residuum** (Soll-Ist-Differenz)
  `res <- resid (o); length(res); mean(res)` — `150 7.519834e-18`
- **Modellparameter** (Versatz & Termgewichte)
  `coef (o)`

(Intercept)	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
0.11822289	0.06602977	0.24284787	-0.22465712	-0.05747273

- **Akaike Informationskoeffizient** (Vorhersagefehler + Strafterm)
  `AIC (o)` — `-145.8171`
- **Konfidenzintervalle** der Parameterschätzwerte
  `confint (o, level=0.95)`

	2.5%	97.5%
(Intercept)	-0.150513257	0.38695904
Sepal.Length	-0.009598443	0.14165798
Sepal.Width	0.164540609	0.32115514
Petal.Length	-0.299240548	-0.15007368
Petal.Width	-0.181393910	0.06644845

Modellklassen und ihre Standardmethoden

... für Fortgeschrittene ...

- **Klassische Varianzanalyse** (ANOVA)

`anova (o)`

```
                     Response: Species == "setosa"
              Df    Sum Sq   Mean Sq    F value    Pr(>F)
Sepal.Length   1   17.1562   17.1562   811.1458    <2e-16   ***
Sepal.Width    1    9.1046    9.1046   430.4664    <2e-16   ***
Petal.Length   1    3.9880    3.9880   188.5511    <2e-16   ***
Petal.Width    1    0.0178    0.0178     0.8402    0.3608
Residuals    145    3.0668    0.0212
```

Signifikanzkodierung: $\underbrace{0 \quad 0.001}_{***}\underbrace{\quad 0.01}_{**}\underbrace{\quad 0.05}_{*}\underbrace{\quad 0.1}_{.}\underbrace{\quad 1}_{}$

- **Kovarianzanalyse** zwischen Parameterschätzwerten

`vcov (o)`

	(Intercept)	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
(Intercept)	0.018487	-0.002717	-0.001637	0.000374	0.000944
Sepal.Length	-0.002717	0.001464	-0.000952	-0.001038	0.000814
Sepal.Width	-0.001637	-0.000952	0.001569	0.000919	-0.000875
Petal.Length	0.000374	-0.001038	0.000919	0.001423	-0.002060
Petal.Width	0.000944	0.000814	-0.000875	-0.002060	0.003931

Verallgemeinertes lineares Vorhersagemodell

Linearkombination ⇝ Gelenkfunktion ⇝ Fehlerverteilung

Link-Funktion zur Quelle-Ziel-Koppelung

Lineares Modell für $\mathrm{g}(y)$ statt für y selbst:

$$\mathrm{g}(\mathbb{Y}|_{\boldsymbol{x}}) = h(\boldsymbol{x}, \boldsymbol{a}) + \mathbb{E} = a_0 + \sum_{j=1}^{n} a_j \cdot x_j + \mathbb{E} \qquad \begin{cases} a_0 & h(\mathbf{0}, \boldsymbol{a}) \\ a_j & x_j\text{-Steigung} \\ \mathbb{E} & \sim??? \end{cases}$$

Inverse Link-Funktion zur Parameterschätzung

Finde Optimalwerte $\boldsymbol{a}$ unter postulierter Verteilungsannahme für

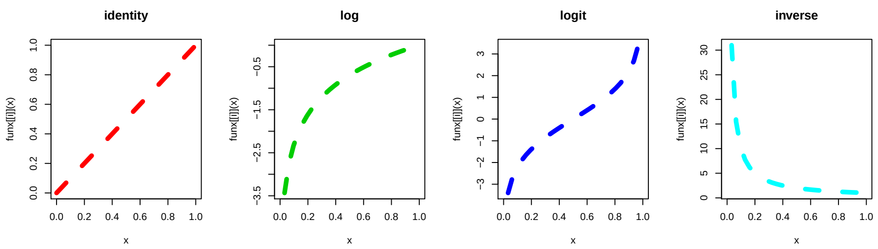
$$\mathbb{Y}|_{\boldsymbol{x}} = \mathrm{g}^{-1}\left(a_0 + \sum_{j=1}^{n} a_j \cdot x_j\right)$$

Bemerkung

OLS ist Spezialfall mit $\mathrm{g}(y) = y$ und $\mathcal{N}(\sigma^2)$.
Nur OLS besitzt **geschlossene Lösung** für $\boldsymbol{a}_{\mathrm{ML}}$.

Verallgemeinertes lineares Modell

Typische Annahmekonfigurationen für Link & Verteilung



Identität

Normal-Verteilung

$$\mathrm{g}(y) = y$$
$$\mathrm{g}^{-1}(\mu) = \mu$$

Logarithmus

Poisson-Verteilung

$$\log(y)$$
$$e^{\mu}$$

Logit

Binomial-Verteilung

$$\log\left(\frac{y}{1-y}\right)$$
$$\frac{e^{\mu}}{1+e^{\mu}}$$

Reziprok

Gamma-Verteilung

$$\mathrm{g}(y) = \frac{1}{y}$$
$$\mathrm{g}^{-1}(\mu) = \frac{1}{\mu}$$

Verallgemeinertes lineares Modell in R

`glm (formula, family=gaussian, data, weights, subset, contrasts, ...)`

Modell & Lerndaten

Modellformel	`formula`
Datensatz	`data`
Fallgewichte	`weights`
Fallauswahl	`subset`

IRLS-Algorithmus

`start`, `etastart`, `mustart`
`method`, `control`

Fehlerdichte & Link

`family` $\in$ { `gaussian`, `binomial`, `Gamma`, `poisson`, `inverse.gaussian`, `quasi`, `quasibinomial`, `quasipoisson` }

Beispiel: Iris-Spezies `setosa`

```
o <- glm (Species=="setosa" ~ .,
          family=binomial,
          data=iris)
```

```
Coefficients:
Intercept  Sep.Len  Sep.Wid  Pet.Len  Pet.Wid
-16.946    11.759   7.842    -20.088  -21.608
```

`predict (o, type="link")`

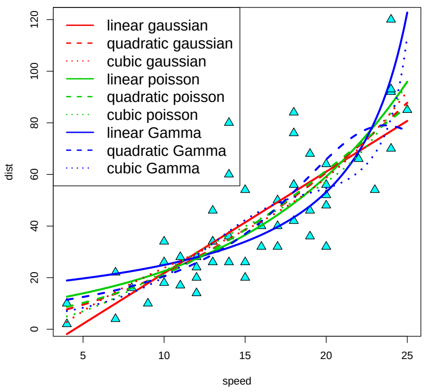
```
38.03 31.75 32.98 27.00 ... -75.55 -65.38
```

`predict (o, type="response")`

```
1 1 1 1 1 1 1 1 1 1 1 ... 0 0 0 0 0
```

Beispiel — Bremsweg aus Geschwindigkeit schätzen

Datensatz 'cars' mit 50 Fällen (speed [mph], dist [ft]) aus den 1920ern



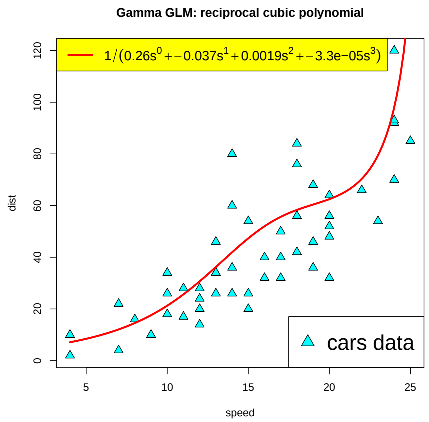

```
attach (cars)
plot (dist~speed, pch=24, bg="cyan", cex=1.4)

spd <- seq (min(speed), max(speed), len=100)
frms <- list (
    linear=dist~speed,
    quadratic=dist~speed + I(speed^2),
    cubic=dist~speed + I(speed^2) + I(speed^3)
    )
fams <- list ("gaussian", "poisson", "Gamma")

for (i in seq(along=frms))
    for (j in seq(along=fams)) {
        o <- glm (
            formula=frms[[i]],
            family=fams[[j]])
        guess <- predict (o,
            newdata=list (speed=spd),
            type="response")
        lines (spd, guess, lty=0+i, col=1+j, lwd=3)
        }
legend ("topleft",
    lty=rep (0+seq(along=frms), times=length(fams)),
    col=rep (1+seq(along=fams), each=length(frms)),
    legend=paste (
        rep (names(frms), times=length(fams)),
        rep (fams, each=length(frms)))
    )
```

Vorhersagegüte

AIC	gaussian	poisson	Gamma
linear	419.1	524.0	427.3
quadratic	418.7	515.6	418.6
cubic	419.8	506.1	412.4

Beispiel — das erfolgreichste (AIC) Bremsweg-Modell

Gamma-Verteilung · reziproke Linkfunktion · Polynom dritten Grades (in speed)




```
o <- glm (
    dist ~ speed+I(speed^2)+I(speed^3),
    family=Gamma, data=cars)
cf <- signif (coef(o), 2)
poly <- paste (cf, "*s^", 0:3, collapse="+")
body <- paste ("1/(", poly, ")")
decl <- paste ("fun <- function(s)", body)
eval (parse (text=decl))

plot (dist ~ speed, data=cars,
    pch=24, bg="cyan", cex=1.4)
curve (fun, add=TRUE, lwd=3, col="red")
legend ("topleft",
    legend=body (fun),
    bg="yellow", lty=1, col="red")
```

`coef (o)` ⇨

(Intercept)	speed	I(speed^2)	I(speed^3)
2.6037e-01	-3.6677e-02	1.8853e-03	-3.2829e-05

`confint (o, level=0.95)` ⇨

	2.5 %	97.5 %
(Intercept)	1.544566e-01	3.810079e-01
speed	-5.848412e-02	-1.700746e-02
I(speed^2)	7.085221e-04	3.161512e-03
I(speed^3)	-5.691372e-05	-1.021716e-05

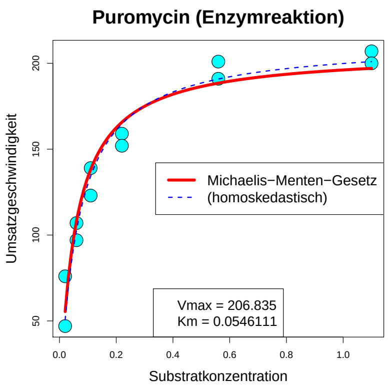
Nichtlineare Modelle in R

`nls (formula, data, start, control, algorithm, subset, weights, ...)`

Michaelis-Menten-Kinetik für Enzymreaktionen

Reaktionsgeschwindigkeit ⇦ Konzentration, Konstanten V_{max}, K_m

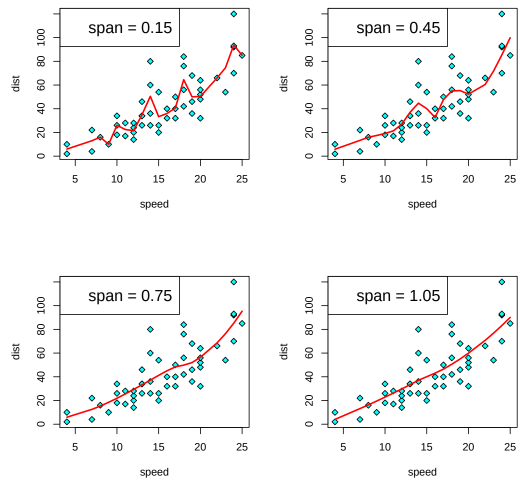
$$y_{rate} = (V_{max} \cdot x_{con})/(K_m + x_{con}) \quad \text{und} \quad \boxed{\sigma_y \propto \sqrt{y_{rate}}}$$




```
Treated <- subset (Puromycin, state=="treated")

plot (rate ~ conc, data=Treated,
    pch=21, cex=3, bg="cyan")

WMM <- function (rate, conc, Vmax, Km) {
    guess <- (Vmax * conc) / (Km + conc)
    (rate - guess) / sqrt (guess)
    }

o <- nls (0 ~ WMM (rate, conc, Vmax, Km),
    data=Treated,
    start=list (Vmax=200, Km=0.1))

cf <- coef (o)
for (i in seq(along=cf))
    assign (names(cf)[i], cf[i])
curve ((Vmax*x)/(Km+x), add=TRUE, lwd=5, col="red")
```


Lokale polynomiale Regression

`loess (formula, data, weights, subset, span=0.75, degree=2, ...)`



Lokale Regression

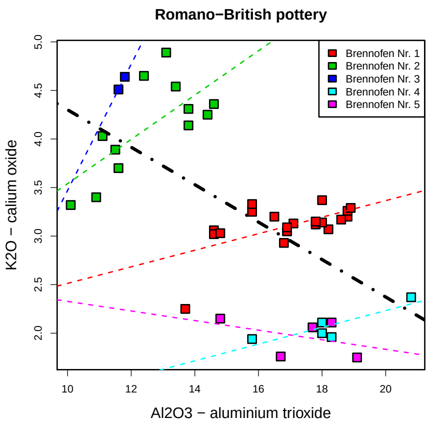
Polynomgrad 0,1,2 `degree`
Nachbarschaft `span`
Daten `formula data`
maximal 4 Quellvariable !

```
for (sp in seq (.15, 1.1, .3))
    {
    plot (dist~speed,
        data=cars,
        pch=23, bg="cyan")
    lines (cars$speed,
        predict (loess (
            dist~speed,
            data=cars,
            span=sp)),
        lwd=2, col="red")
    }
```

LOcally WEighted Scatterplot Smoothing (Vanille-Variante):

`lowess (x, y=NULL, f=2/3, iter=3, delta=0.01*diff(range(xy$x[o])))`

Faktoren als Quellattribute




```
library (HSAUR2); attach (pottery)

plot (K2O ~ Al2O3,
   bg=1+unclass (pottery$kiln))
abline (lm (K2O ~ Al2O3), lty=4, lwd=5)
for (i in seq(along=levels(kiln)))
   abline (lm (K2O ~ Al2O3,
      subset=unclass(kiln)==i), lty=2, col=1+i)
```

Pottery-Datensatz

45 Fundstücke
5 Brennöfen kiln
9 Chemikalien:

Al2O3	aluminium trioxide
Fe2O3	iron trioxide
MgO	magnesium oxide
CaO	calcium oxide
Na2O	natrium oxide
K2O	calium oxide
TiO2	titanium oxide
MnO	mangan oxide
BaO	barium oxide

Akaike-Information

Vorhersage von K_2O aus

Al_2O_3 allein	97.30
Al_2O_3 und kiln	16.11
kiln allein	29.11

Vorhersageformel für gemischte Attributskalen?

Faktoren und Vorhersageformeln

Faktorattribut $k \in \{1, 2, \ldots, \ell\}$ ➪ numerische Hilfsattribute $k_1, \ldots, k_\ell \in \mathbb{R}$

Beispiel Al_2O_3-Vorhersage (pottery-Daten)

$y =$		K2O	kiln	K2O+kiln	K2O:kiln	K2O*kiln
	$1\cdot$ Intercept	21.9	16.9	8.68	6.04	4.44
+	$x\cdot$ K2O	-1.94		2.66		4.02
+	$k_2\cdot$ kiln2		-4.36	-7.07		-0.93
+	$k_3\cdot$ kiln3		-5.22	-9.13		0.23
+	$k_4\cdot$ kiln4		1.26	3.99		-4.27
+	$k_5\cdot$ kiln5		0.40	3.42		19.89
+	$x\cdot k_1\cdot$ K2O:kiln1				3.50	
+	$x\cdot k_2\cdot$ K2O:kiln2				1.59	-1.83
+	$x\cdot k_3\cdot$ K2O:kiln3				1.24	-2.48
+	$x\cdot k_4\cdot$ K2O:kiln4				5.86	4.66
+	$x\cdot k_5\cdot$ K2O:kiln5				5.66	-7.59
AIC		201.1	173.8	160.8	162.3	159.4

Bemerkung
Überflüssige Hilfsvariable (lineare Abhängigkeit!) werden automatisch getilgt.

Welche Werte bekommen die Hilfsvariablen?

`model.matrix (object, data=environment(object), contrasts.arg=NULL, ...)`

Ohne Interaktionen zwischen kiln und K2O

`model.matrix (Al2O3~K2O+kiln, pottery)`

	(Intercept)	K2O	kiln2	kiln3	kiln4	kiln5
1	1	3.20	0	0	0	0
2	1	3.05	0	0	0	0
3	1	3.07	0	0	0	0
22	1	4.25	1	0	0	0
23	1	4.14	1	0	0	0
34	1	4.51	0	1	0	0
36	1	1.96	0	0	1	0
41	1	2.06	0	0	0	1
45	1	1.75	0	0	0	1

Mit Interaktionen zwischen kiln und K2O

`model.matrix (Al2O3~K2O:kiln, pottery)`

	(Intercept)	K2O:kiln1	K2O:kiln2	K2O:kiln3	K2O:kiln4	K2O:kiln5
1	1	3.20	0.00	0.00	0.00	0.00
2	1	3.05	0.00	0.00	0.00	0.00
3	1	3.07	0.00	0.00	0.00	0.00
22	1	0.00	4.25	0.00	0.00	0.00
23	1	0.00	4.14	0.00	0.00	0.00
34	1	0.00	0.00	4.51	0.00	0.00
36	1	0.00	0.00	0.00	1.96	0.00
41	1	0.00	0.00	0.00	0.00	2.06
45	1	0.00	0.00	0.00	0.00	1.75

Faktoren und Kontrastmatrizen

`contrasts (x, contrasts=TRUE, sparse=FALSE)` — zum Abfragen und Setzen

`contrasts (pottery$kiln)`

	2	3	4	5
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1

`contrasts (factor (LETTERS[1:6]))`

	B	C	D	E	F
A	0	0	0	0	0
B	1	0	0	0	0
C	0	1	0	0	0
D	0	0	1	0	0
E	0	0	0	1	0
F	0	0	0	0	1

`contrasts (Windrose) <- "contr.treatment"`

	west	sued	ost
nord	0	0	0
west	1	0	0
sued	0	1	0
ost	0	0	1

`contrasts (Windrose) <- "contr.helmert"`

	[,1]	[,2]	[,3]
nord	-1	-1	-1
west	+1	-1	-1
sued	0	+2	-1
ost	0	0	+3

`contrasts (Windrose) <- "contr.sum"`

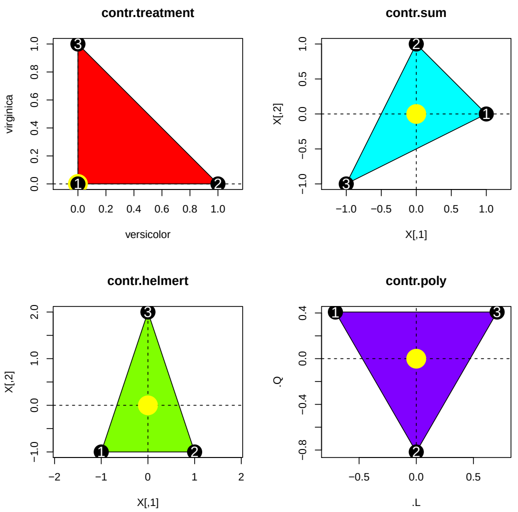
	[,1]	[,2]	[,3]
nord	1	0	0
west	0	1	0
sued	0	0	1
ost	-1	-1	-1

`contrasts (Windrose) <- "contr.poly"`

	.L	.Q	.C
nord	-0.6708204	0.5	-0.2236068
west	-0.2236068	-0.5	0.6708204
sued	0.2236068	-0.5	-0.6708204
ost	0.6708204	0.5	0.2236068

Geometrie von ℓ Punkten des $\mathbb{R}^{\ell-1}$

Äquidistante Codekonfiguration durch orthogonale Polynome




```
contr <- paste ("contr", c(
    "treatment",
    "helmert",
    "sum",
    "poly"
    ), sep=".")
attach (iris)
layout (matrix (1:4, 2, 2))
for (i in seq(along=contr)) {
    contrasts (Species) <- contr[i]
    X <- contrasts (Species)
    plot (X, asp=1, main=contr[i])
    polygon (X, col=rainbow(4)[i])
    abline (h=0, v=0, lty=2)
    points (0, 0,
       cex=4, col="yellow", pch=19)
    points (X, pch=19, cex=3)
    text (X, labels=1:3,
       col="white", cex=1.4)
    }
```


Faktoren als Zielattribute

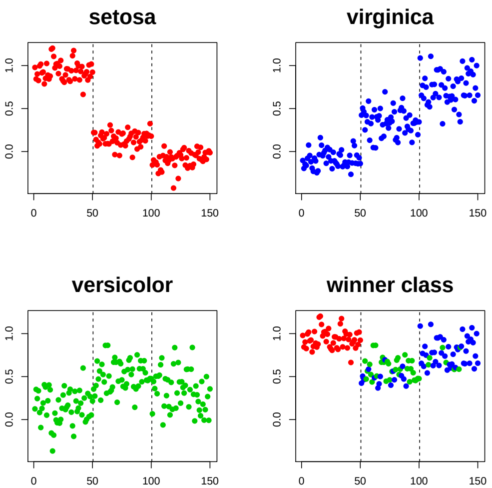
Verwendung von Vorhersagemodellen zu Klassifikationszwecken

- **Faktoren können nicht Zielvariable sein!**
  `lm (Species ~ ., data=iris)` `Fehler in storage.mode(y) <- "double" ...`
- **Kontrastmatrix basteln**
  `Y <- diag (length (levels (Species)))` Einheitsvektoren ($\mathbb{R}^3$)
  `Y <- contr.helmert (levels (Species))` Helmertkontraste ($\mathbb{R}^2$)
- **Zielmatrix erzeugen**
  `y <- Y[Species,]` je Faktoreintrag passende Kontrastzeile ($\mathbb{R}^{150\times 3}$)
- **Multiples Vorhersagemodell**
  `o <- lm (y ~ . - Species, data=iris)`

```
                   [,1]       [,2]       [,3]
(Intercept)    0.118223   1.577059  -0.695282
Sepal.Length   0.066030  -0.020154  -0.045876
Sepal.Width    0.242848  -0.445616   0.202768
Petal.Length  -0.224657   0.220669   0.003988
Petal.Width   -0.057473  -0.494307   0.551779
```


Beispiel — lineare OLS-Klassifikation

150 Irisblüten $=$ 50$\times$ `setosa` $+$ 50$\times$ `versicolor` $+$ 50$\times$ `virginica`




```
attach (iris)
y <- diag(3)[Species,]
o <- lm (y~.-Species, data=iris)
u <- predict (o, newdata=iris)

par (pch=19, cex.main=2)
layout (matrix (1:4, 2, 2))

newplot <- function (S, brx=NULL, ...) {
   plot (c(1,nrow(S)), range(S),
      xlab="", ylab="", type="n", ...)
   abline (v=1/2+brx, lty=2)
   }

clab <- levels (Species)
for (k in seq(along=clab)) {
   newplot (u, c(50,100), main=clab[k])
   points (u[,k], col=1+k)
   }

u.max <- apply (u, MARGIN=1, max)
k.max <- apply (u, MARGIN=1, which.max)
newplot (u, c(50,100), main="winner class")
points (u.max, col=1+k.max)
```


Weitere Vorhersagemodelle

- **Korrelierte Vorhersagefehler**
  Generalized least squares `nlme::gls()`
- **Effekte von Quellvariablen auf Parameter**
  Linear mixed-effects model `nlme::lme()`
  Non-linear mixed-effects model `nlme::nlme()`
- **Verallgemeinerte lineare Modelle**
  Splineregression, automatische Glättung `mgcv::gam()`
- **Vorhersage mit neuronalen Netzen**
  Single hidden layer perceptron `nnet::nnet()`
- **Vorhersage geordneter Faktoren**
  Proportional-odds logistic regression `MASS::polr()`

Bemerkung

Mehr Modelle zur Vorhersage von Faktoren im Kapitel zur „*Klassifikation*“

Funktionen für Wahrscheinlichkeitsverteilungen

Generelle Statistikfunktionen

Stetige univariate Verteilungen

Nichtnegative univariate Verteilungen

Univariate Verteilungen auf $[0,1]$

Diskrete (univariate) Verteilungen

Formelschnittstelle für Vorhersagemodelle

Lineare und nichtlineare Modelle

Numerische Optimierung

Verteilungen Statistik reelle WV positive WV [0,1]-WV diskrete WV ?formula N/L-Modelle **Optimierung**

Was ist Optimierung?

Spezielle Form der inversen Aufgabenstellung zur Funktionsauswertung

$$f : \begin{cases} \mathcal{M} & \to & \mathbb{R} \\ x & \mapsto & f(x) \end{cases} \qquad \begin{array}{ll} f(x^*_{\min}) \le f(x) & (\forall x \in \mathcal{M}) \\ f(x^*_{\max}) \ge f(x) & (\forall x \in \mathcal{M}) \end{array}$$

Suchraum $\hat{=}$ Definitionsbereich

Diskrete/kombinatorische Optimierung	$\lvert\mathcal{M}\rvert < \infty$
Skalare (univariate) Optimierung	$\mathcal{M} = \mathbb{R}$
Mehrdimensionale (multivariate) Optimierung	$\mathcal{M} = \mathbb{R}^n$

Suchstrategie

Traversieren des Raums
Geordnete Suche (Kand.-Liste)
Evolutionäre Suche

globales vs. lokales Optimum

Suchinformation

Funktionsauswertung	$f(x)$
Gradientenauf/abstieg	$f'(x)$
Schrittweitenbestimmung	$f''(x)$

deterministisch vs. stochastisch

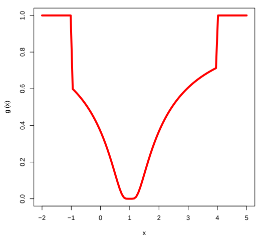
Verteilungen Statistik reelle WV positive WV [0,1]-WV diskrete WV ?formula N/L-Modelle **Optimierung**

Skalare Optimierung (goldener Schnitt)

`optimize (f=, interval=, ..., lower, upper, maximum=FALSE)`

- **Minimum einer Parabel**
  `optimize (f=function(r) (r-3)^2, interval=c(-5,+5))`
  {min=3 obj=0}
- **Maximum der Sinuswelle**
  `optimize (f=sin, interval=c(0,5), maximum=TRUE)`
  {min=1.5708 obj=1.0000}
- **Besetzen überschießender Funktionsargumente**
  `optimize (f=get('*'), interval=c(-5,+5), e2=2)`
  {min=-4.999 obj=-9.999}
  (Abholen des Funktionsobjekts; Namen der Argumente)
- **Wir legen den goldenen Schnitt herein!**

```
g <- function(x) ifelse (
     x>-1, ifelse (
        x<4, exp (-1/abs(x-1)), 1), 1)
```



`optimize(g, c(-7,20))` {min=0.99 obj=0.000}

`optimize(g, c(-4,20))` {min=19.99 obj=1.000}

Verteilungen Statistik reelle WV positive WV [0,1]-WV diskrete WV ?formula N/L-Modelle **Optimierung**

Beispiel — Maximum-Likelihood-Schätzung

Optimale Parameter μ, σ einer normalverteilten Datenprobe

- **Erzeuge univariate Datenprobe**
  `x <- rnorm (100, mean=13, sd=3)`
- **Definiere Likelihoodfunktion**
  ```
  like <- function (mu=0, sd=1, data=x)
              sum (dnorm (data, mu, sd, log=TRUE))
  ```
- **Maximiere hinsichtlich** `mu`
  `optimize (like, c(0,100), maximum=TRUE)`
  `maximum=12.9195 objective=-699.9`
- **Maximiere hinsichtlich** `sd`
  `optimize (function(s) -like (sd=s), c(0,100))`
  `minimum=13.38 objective=401.3`
- **Verwende geschätzten** `mu`**-Wert**
  `optimize (function(s) -like (mu=12.9, sd=s), c(0,100))`
  `minimum=3.48 objective=266.8`

Univariate Nullstellensuche

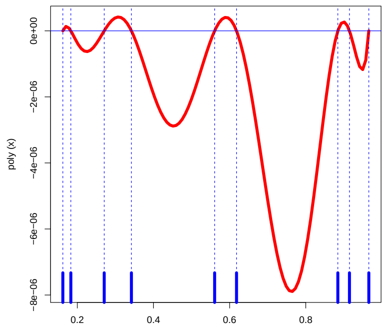
```
uniroot (f, interval, ..., lower, upper, f.lower, f.upper, maxiter=1000)
```

- **Intervallangabe erforderlich**
  ```
  uniroot (f=cos, interval=c(-1,+2))
  root=1.57 f.root=1.2e-07 iter=5
  ```
- **Intervallgrenzen nur mit Vorzeichenwechsel!**
  ```
  uniroot (f=cos, lower=-1, upper=+1)
  f() values at end points not of opposite sign
  ```
- **Nur eine Nullstelle wird gesucht**
  ```
  uniroot (function(x) (x-2)*(x-4), c(0,3))
  root=2 f.root=-1.57e-05 iter=8
  ```
- **Überschießende Funktionsargumente**
  ```
  uniroot (function(x,z) x*x-z, c(1,2), z=2)
  root=1.41 f.root=-6.86e-07 iter=5
  ```
- **Argumentnamen aus Fehlermeldung**
  ```
  uniroot (f='-', c(-1,+5), a=3)
  root=3 f.root=0 iter=1
  ```


(Komplexe) Polynomwurzeln (Jenkins & Traub 1972)

`polyroot (z)` mit $p(x) = z_1 + z_2 \cdot x + z_3 \cdot x^2 + \ldots + z_n \cdot x^{n-1}$

- **Quadratisches Polynom** z.B. $1 - 3x + x^2$ oder $1 + x^2$
  ```
  polyroot (c(1,-3,1))          0.381966-0i 2.618034+0i
  polyroot (c(1,0,1))                         0-1i 0+1i
  ```
- **Vielfache Nullstellen** z.B. $(1+x)^8$
  ```
  choose(8, 0:8)                1 8 28 56 70 56 28 8 1
  polyroot (choose(8, 0:8)) -1-0i -1+0i -1+0i -1-0i -1-0i -1-0i -1-0i -1-0i
  ```




```
a <- runif (9)
poly <- function (x)
   apply (outer(x,a,"-"), 1, prod)
plot (poly, min(a), max(a),
   col="red", lwd=5)
abline (v=a, lty=2, col="blue")
abline (h=0, lty=1, col="blue")
rug (
   Re (polyroot (polyCF (a))),
   col="blue", lwd=5,
   ticksize=0.1)
```


Multivariate Optimierung — Newton-Verfahren

```
nlm (f, p, ..., hessian=FALSE, gradtol=1e-6, iterlim=100)
```

- **Erzeuge univariate Datenprobe**
  ```
  x <- rnorm (100, mean=13, sd=3)
  ```
- **Definiere negative Likelihoodfunktion**
  ```
  neglike <- function (para, data=x)
     -sum (dnorm (data, para[1], para[2], log=TRUE))
  ```
- **Minimiere simultan hinsichtlich** para $= (\mu, \sigma)$
  ```
  nlm (f=neglike, p=c(10,2))
  minimum=250.5011 estimate=12.999695/2.962615 code=1 iterations=28
  ```

Newton-Abstieg benötigt Gradient & Hessematrix

implizit: $\nabla_p f$ und $H_p f$ werden von `nlm` numerisch angenähert.
explizit: Berechnungsfunktionen werden als Attribute `gradient` und `hessian` von `f` übergeben.

Multivariate Minimierung mit Nebenbedingungen

```
nlminb (start, objective, gradient, hessian, ..., lower=-Inf, upper=Inf)
```

- **Maximum-Likelihood (wie `nlm`)**
  ```
  nlminb (start=c(0,1), objective=neglike)
  par=12.9997/2.9626 obj=250.501 iter=16 eval/fun=20 eval/grad=46
  ```
- **Maximale Spannweite gesucht**
  ```
  foo <- function (z) -abs (diff (range (z)))
  ```
- **Pathologisch ohne Schranken:**
  ```
  nlm (f=foo, p=1:5)        -26221.22 2.00 3.00 4.00 +26227.22
  nlminb (start=1:5, objective=foo)
       -5.1111e+11 2.0000e+00 3.0000e+00 4.0000e+00 5.1111e+11
  ```
- **Wohldefiniert mit Schranken:**
  ```
  nlminb (start=1:5, obj=foo, lower=-883, upper=+4711)
                                              -883 2 3 4 4711
  ```
- **Fokussierung bei relativen Minima** $f(z) = \sin(z_1) + \cos(z_2)$
  ```
  foo <- function(z) sin(z[1])+cos(z[2])
  nlminb (c(35,35), foo, lo=33, up=37)
                                       par=36.128/34.557 obj=-2
  ```

Multivariater Allzweckminimierer

`optim (par, fn, gr=NULL, ..., method, lower, upper, control=list())`

- **Rosenbrocks Bananenfunktion**

```
frb <- function(x)
    100 * (x[2]-x[1]^2)^2 + (1-x[1])^2
```

- **... und ihre ersten Ableitungen**

```
grb <- function(x) c(
    -400 * x[1] * (x[2]-x[1]^2) - 2 * (1-x[1]),
    200 * (x[2]-x[1]^2))
```

- **Kein Erfolg mit Nelder-Mead**

```
optim (c(-1.2,1), frb)
    par=0.9998044/0.9996084 value=3.827383e-08
```

- **BFGS mit Gradienten**

```
optim (c(-1.2,1), frb, grb, method='BFGS')
    par=1/1 value=9.594956e-18
```

Methoden

`Nelder-Mead` nur f-Werte

`BFGS` Broyden, Fletcher, Goldfarb & Shanno; f und f'

`CG` Konjugierte Gradienten (HDim)

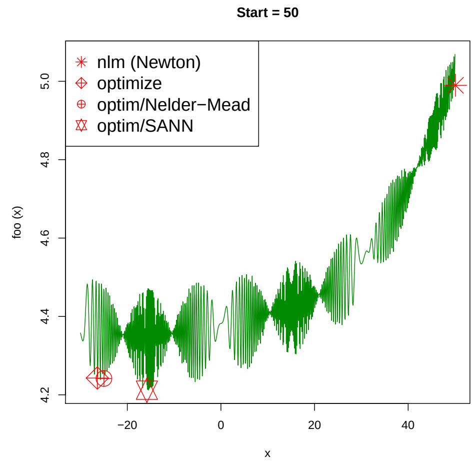
`SANN` Simulated Annealing

`BFGS` BFGS + Schranken

Skalare Optimierung — viele lokale Minima

Simulated Annealing mit 10000 Auswertungen und handverlesener Abkühlpolitik

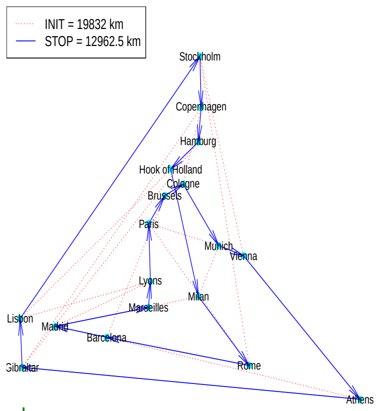
$$f(x) \;=\; \log\left\{10 \cdot \sin(0.3x) \cdot \sin(1.3x^2) + \frac{x^4}{10^5} + \frac{x}{5} + 80\right\}$$




```
foo <- function (x) log (
       10*sin(0.3*x)*sin(1.3*x^2) +
       0.00001*x^4 + 0.2*x + 80
       )
plot (foo, -30, +50, n=1000,
       col="green4", main="Start = 50")

o <- nlm (f=foo, p=50)
points (o$estimate, o$minimum,
       col="red", pch=8, cex=3)
o <- optimize (f=foo, interval=c(-50,+50))
points (o$minimum, o$objective,
       col="red", pch=9, cex=3)
o <- optim (par=50, fn=foo, method="Nelder-Mead")
points (o$par, o$value,
       col="red", pch=10, cex=3)
o <- optim (par=50, fn=foo, method="SANN",
       control=list(parscale=20))
points (o$par, o$value,
       col="red", pch=11, cex=3)
```


Travelling Salesperson Problem (TSP)

Näherungslösung durch *„Simulated Annealing“*



Bemerkung

Die `SANN`-Methode von `optim` erwartet als Argument `gr` keine Funktion für den **Gradienten**, sondern für den **Folgekandidaten**.

```
distance <- function (s, P=xy)
   sum (as.matrix (dist (P))[embed(s,2)])

newseq <- function (s, P=xy) {
   idx <- 3:nrow(P) - 1
   changepoints <- sample (idx, size=2)
   s[changepoints] <- rev (s[changepoints])
   s}

idx <- 1:nrow(xy)
s <- c (idx, 1)
tsp <- xy[s,]
plot (xy, asp=1,
   axes=FALSE, pch=19, col="cyan")
arrows (tsp[idx,1], tsp[idx,2],
   tsp[idx+1,1], tsp[idx+1,2],
   lty=3, length=0, angle=10, col="red")

set.seed (883)
o <- optim (s, distance, newseq, method="SANN")
tsp <- xy[o$par,]
arrows (tsp[idx,1], tsp[idx,2],
   tsp[idx+1,1], tsp[idx+1,2],
   lty=1, length=0.2, angle=10, col="blue")

text (xy, rownames (xy), cex=0.8)
```


Zusammenfassung (4)

1. Mit `mean`, `var` etc. berechnen wir (getrimmte) **Mittel** und **Streuungen** bzw. **Kovarianzen** von Datenproben.
2. **Rangordnungen** werden mit `sort`, `rank`, `order` analysiert, **Quantile** mit `quantile` und `cut`.
3. Verteilungen von Datenproben werden mit `qqplot` **verglichen** und mit `qqnorm` auf **Normalität** getestet.
4. Für alle gängigen Wahrscheinlichkeitsmodelle bietet 'R' Methoden [`d`,`p`,`q`,`r`]`xxx` für den **Dichteverlauf**, für die **kumulative Verteilung**, für die **Quantilberechnung** und für ein zufallsgesteuertes **Auswürfeln**.
5. **Modellformeln** der Art $V_{\text{resp}} \sim V_{\text{expl}}$ bieten eine kompakte mengentheoretische Notation (`+`, `-`, `:`, `*`, `^`) für die **Interaktionsterme** statistischer **Vorhersagemodelle**.
6. Die Modelle liefern **Schätzwerte** und **Konfidenzintervalle** ihrer Parameter, Qualitätsaussagen wie **AIC**, **ANOVA** und **Residuen** und sie unterstützen die **induktive Prädiktion**.
7. Die Modelle sind **linear** (`lm`), **gekoppelt** linear (`glm`), **nichtlinear** (`nls`) oder **nichtparametrisch** (`loess`).
8. **Diskrete** Quell- oder Zielvariable (*„Faktoren“*) werden mittels **Kontrastmatrizen** konvertiert.
9. **Optimierungsaufgaben** werden mit `optimize` (univariat), `uniroot` und `polyroot` (Nullstellen), `nlm`[`inb`] (Newton multivariat) oder `optim` (zerklüftet, kombinatorisch) gelöst.