

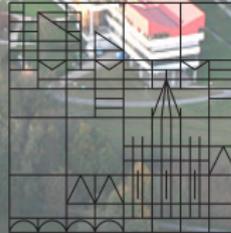
Computerphysik II

Kapitel 1 - Statistik und Datenanalyse

S. Gerlach

WiSe 2022/23

Universität
Konstanz



INHALT

1. Kombinatorik
2. Diskrete Wahrscheinlichkeitsverteilungen
3. Numerische Integration
4. Kontinuierliche Wahrscheinlichkeitsverteilungen
5. Schätzmethoden und Anpassung von Funktionen

Permutation:

Anordnung von n Elementen

$\{1, 2, 3\}$: 123, 132, 321, 213, 231, 312, 321

Anzahl:

$$P_n = n!$$

Anzahl mit Wdh.:

$$P_n = \frac{n!}{n_1! n_2! \cdots n_k!}$$

Berechnung:

$n! = 1 \cdot 2 \cdots n$ (iterativ)

$n! = n(n-1)!$ (rekursiv)

`math.factorial(int)`, `scipy.special.perm(n, k)`,

`math.perm(n, k)` [Python 3.8]

Für große n (Stirling-Formel):

$$n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

bzw.

$$\lg(n!) \approx n(\ln n - 1)$$

Anwendungen Fakultät:

- ▶ Binomialkoeffizient (siehe Kombination)
- ▶ Binomischer Lehrsatz
- ▶ Pascalsches Dreieck
- ▶ Exponentialfunktion ($\sum_{n=0}^{\infty} \frac{x^n}{n!}$)
- ▶ Taylorreihe
- ▶ Leibniz-Produktregel
- ▶ ...

Variation: Auswahl k aus n Elementen (mit **Reihenfolge**)

2 aus $\{1, 2, 3\}$: 12, 13, 21, 23, 31, 32

Anzahl:

$$V_{n,k} = \frac{n!}{(n-k)!}$$

Anzahl mit Wdh.:

$$V_{n,k}^w = n^k$$

Kombination: Auswahl von k aus n Elementen (ohne Reihenfolge)

2 aus $\{1, 2, 3\}$: 12, 13, 23

Anzahl:

$$C_{n,k} = \frac{V_{n,k}}{P_k} = \frac{n!}{k!(n-k)!} = \binom{n}{k}.$$

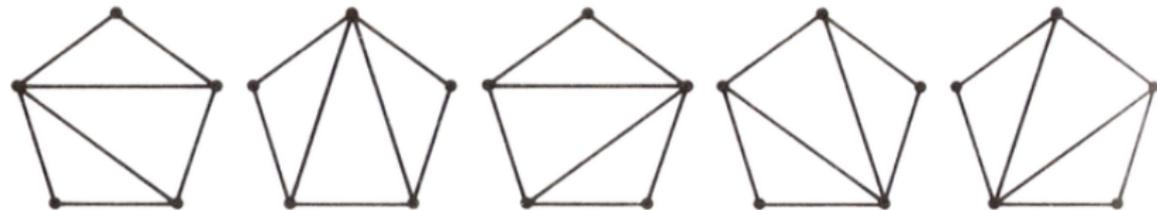
```
1 from math import comb # Python 3.8
2 print ("%d über %d =" % (n, k), comb(n, k, exact=True))
```

```
1 from scipy.special import comb
2 print ("%d über %d =" % (n, k), comb(n, k, exact=True))
```

In Python: `itertools` (s. Übung)

Tipp: `list(...)` erzeugt eine iterierbare Liste, `set` eine Menge (ohne Wiederholungen)

Catalan-Zahlen



<https://de.wikipedia.org/wiki/Catalan-Zahl>

$$C_n = \frac{1}{n+1} \binom{2n}{n}$$

$$C_{n+1} = \frac{2(2n+1)}{n+2} C_n$$

Zuerst: Zufallsexperimente mit diskreter Anzahl an Ergebnissen.

1. Beispiel: **Binomialverteilung**

Mit welcher Wahrscheinlichkeit gibt es k positive Ergebnisse bei n Versuchen eines Bernoulli-Experiments ("Münzwurf")?

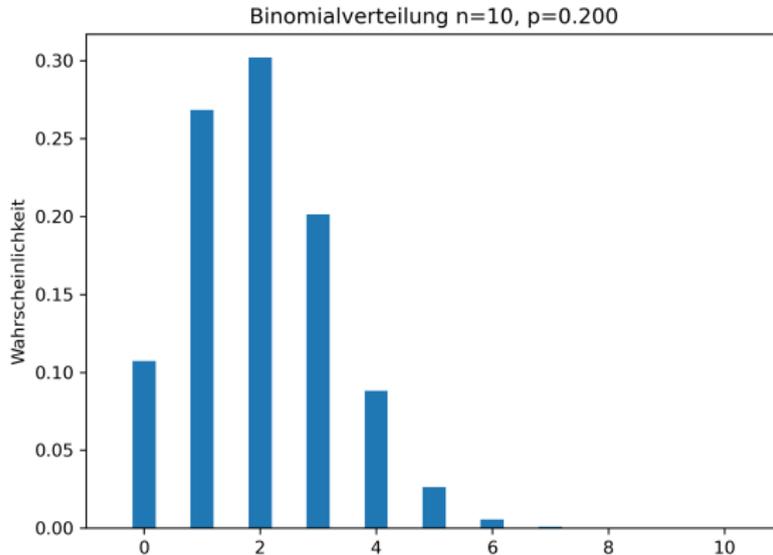
$$P(k) = B_{n,p}(k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Mit $k = 0, \dots, n$. $\binom{n}{k}$ ist die Anzahl der Kombinationen, da egal ist, wann die positiven Ergebnisse auftreten.

Ideale Münze ($p = 1/2$): $P(k) = \binom{n}{k} \frac{1}{2}^k \frac{1}{2}^{n-k} = \frac{1}{2^n} \binom{n}{k} = \frac{C_{n,k}}{V_{n,2}}$

Binomialverteilung

```
1 from scipy.stats import binom
2 import pylab as pl
3
4 k=pl.arange(0, n + 1, 1)
5 pl.bar(k, binom.pmf(k, n, p), width=0.4, align="center")
```

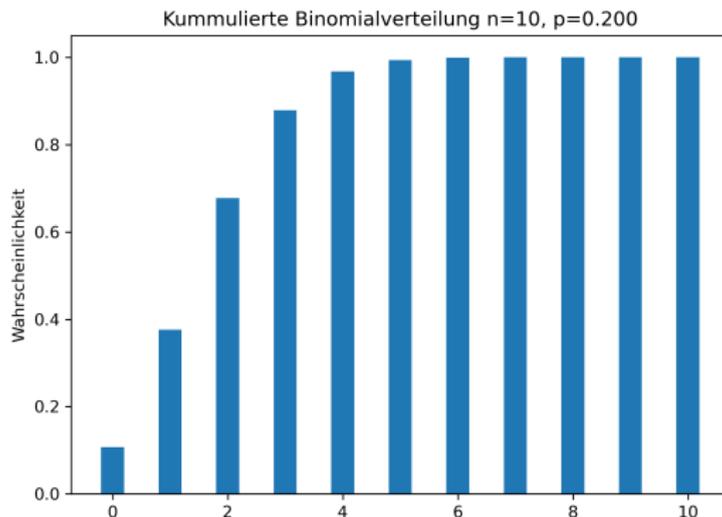


Kumulative Binomialverteilung

Wieviele Ereignisse sind kleiner als ein Wert k_0 : Summierte Wahrscheinlichkeit

$$F_K(k_0) = P(K < k_0) = \sum_{k=0}^{\lfloor k_0 \rfloor} \binom{n}{k} p^k (1-p)^{n-k}$$

1 `pl.bar(k0, binom.cdf(k0, n, p), width=0.4, align="center")`



Eigenschaften:

► **Normierung:**

$$\sum_{k=0}^n P(k) = \sum_{k=0}^n \binom{n}{k} p^k (1-p)^{n-k} \stackrel{\text{binom. Lehrsatz}}{=} (p + (1-p))^n = 1$$

► **Erwartungswert:** $\mu = \langle k \rangle = \sum_{k=0}^n k P(k) = \dots = np$
(= Summe der einzelnen Erwartungswerte)

► **Varianz:** $\sigma^2 = \sum_{k=0}^n (k - \mu)^2 P(k) = \langle (x - \mu^2) \rangle = \dots = \langle x^2 \rangle - \langle x \rangle^2 = \dots = np(1-p)$.

Damit ist die **relative Breite**:

$$\frac{\sigma}{\mu} = \sqrt{\frac{1-p}{np}} \sim \frac{1}{\sqrt{n}}$$

D.h. Mittelwert lässt sich für $n \rightarrow \infty$ beliebig genau bestimmen.

Binomialverteilung - Beispiele

1. Qualitätssicherung:

Situation: Produkte am Fließband sollen kontrolliert werden. Gesucht ist die Fehlerquote p .

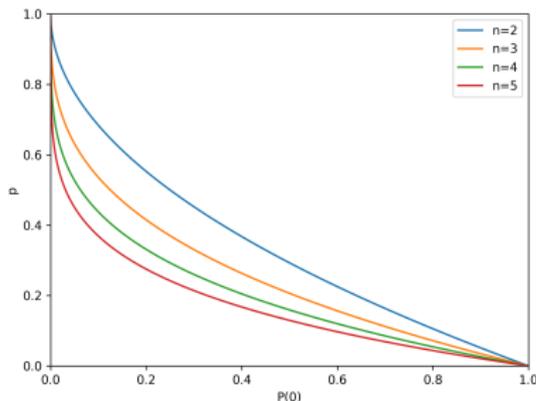
Idee: Messe Wahrscheinlichkeit für Null Fehler $P(0)$ bei Auswahl von jeweils n Teilen:

$$P(0) = B_{n,p}(k=0) = \binom{n}{0} p^0 (1-p)^{n-0} = (1-p)^n$$

Damit

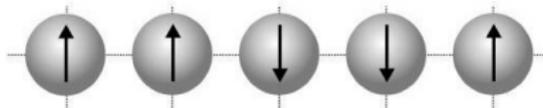
$$p = 1 - \sqrt[n]{P(0)}.$$

Beispiel: $n = 3$, gemessen $P(0) = 0,25 \rightarrow p = 0,37$.



Binomialverteilung - Ising-Modell

Kette von Spins mit Moment $m_i = \pm 1$ mit N_+ (up) und N_- (down)



Magnetisches Moment: $M = \sum_{i=1}^N m_i$
 $M = N_+ \cdot (+1) + N_- \cdot (-1) = N_+ - (N - N_+) = 2N_+ - N$, also

$$N_+ = \frac{M + N}{2}, N_- = \frac{N - M}{2}.$$

$$P(N_+) = \binom{N}{N_+} p^{N_+} (1-p)^{N-N_+} = B_{N,p}(N_+) \quad \text{Binomialverteilung}$$

Damit:

$$\langle M \rangle = \langle 2N_+ - N \rangle = 2\langle N_+ \rangle - N = 2(Np) - N = N(2p - 1).$$

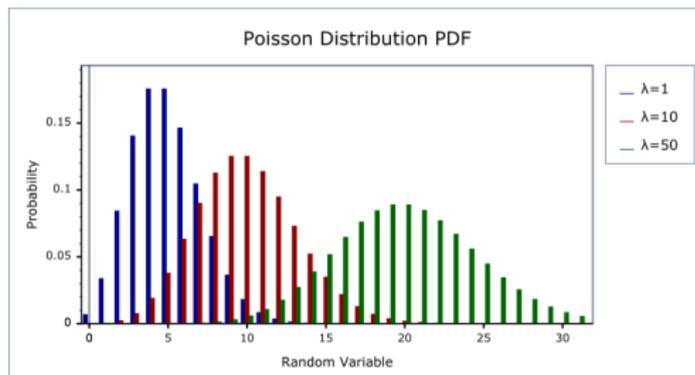
$$\sigma^2 = 4Np(1-p).$$

$$\frac{\sigma}{\langle M \rangle} = \sqrt{\frac{4p(1-p)}{N(2p-1)^2}} \sim \frac{1}{\sqrt{N}}.$$

Poissonverteilung

Grenzfall $n \rightarrow \infty$ bei konstantem Erwartungswert $\lambda = np$ (d. h. $p \rightarrow 0$):

$$P_{\lambda}(k) = \frac{\lambda^k}{k!} e^{-\lambda}$$



Anwendung: viele, aber seltene Ereignisse

```
1 from scipy.stats import poisson
2 pl.bar(k, poisson.pmf(k, l), width=0.4, align="center")
```

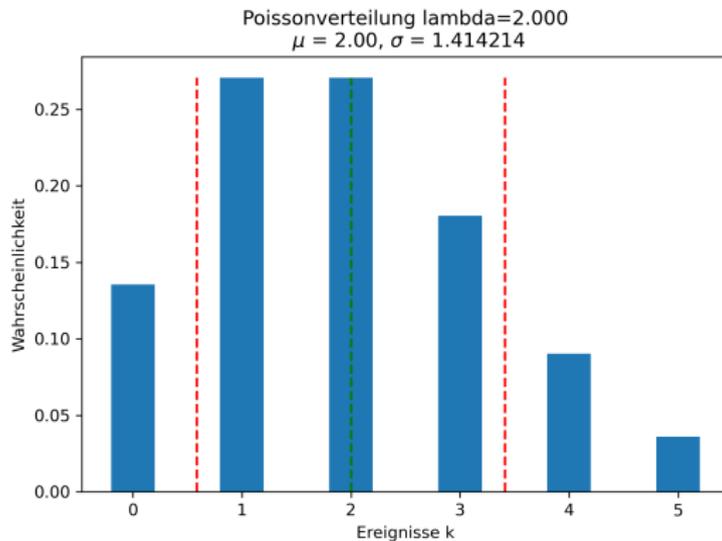
Eigenschaften:

- ▶ Normierung:

$$\sum_{k=0}^n P(k) = \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} e^{-\lambda} = e^{-\lambda} \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} = e^{-\lambda} e^{\lambda} = 1.$$

- ▶ Erwartungswert: $\mu = \sigma^2 = \dots = \lambda$

Relative Breite: $\frac{\sigma}{\mu} \approx \frac{1}{\sqrt{\lambda}}$.

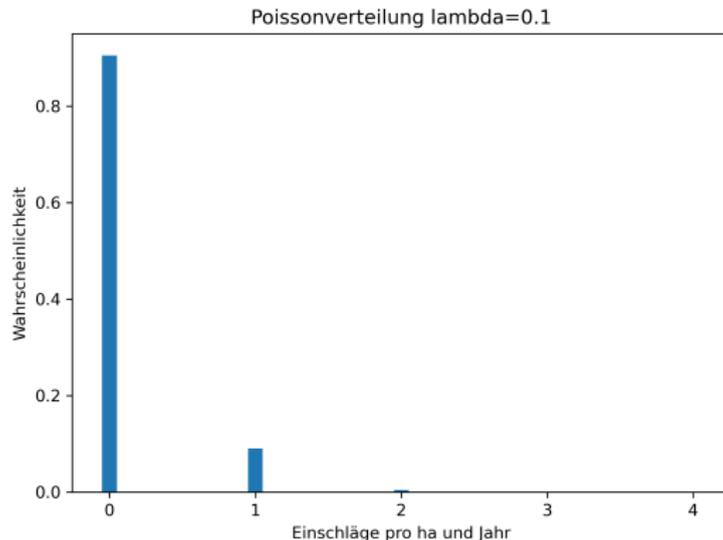


Poissonverteilung - Beispiel 1 - Blitzeinschläge

Im Mittel wurden 0,1 Blitzeinschläge pro Hektar in Deutschland registriert. Mit welcher Wahrscheinlichkeit schlägt kein, ein, zwei, ... Blitze auf einen Hektar ein?

$$\mu = 0,1 = \lambda$$

→ 90% kein Einschlag, 10% ein Einschlag.

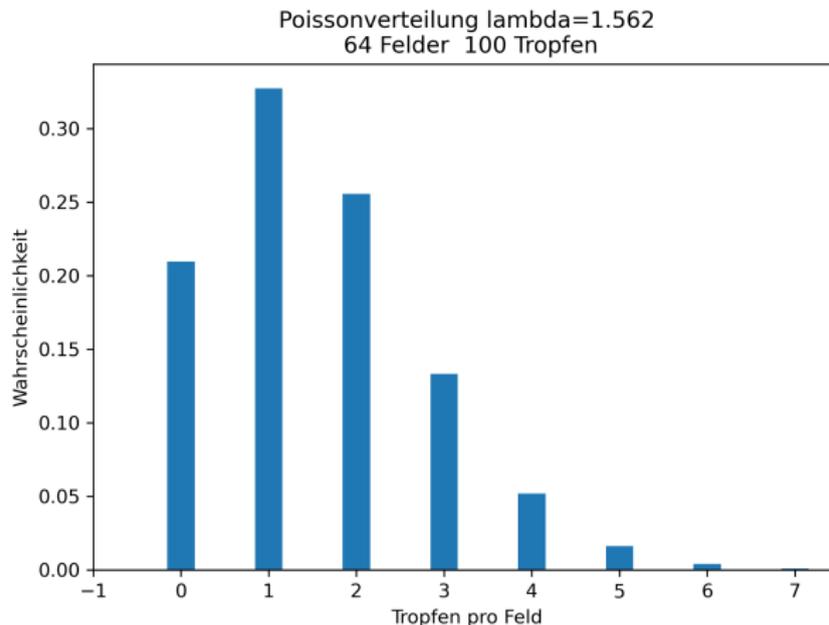


Poissonverteilung - Beispiel 2 - Schachbrett

Es regnet auf ein Schachbrett. Wieviele Tropfen pro Feld erwartet man?

$N = 64$ Felder, $n = 100$ Tropfen: $\lambda = \mu = n/N \approx .1,56$:

Immerhin mit 5% Wahrscheinlichkeit 4 Regentropfen im Feld.



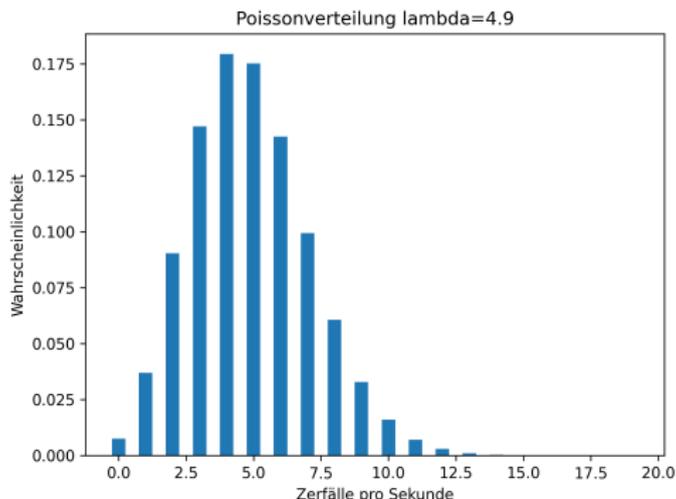
Poissonverteilung - Beispiel 3 - Radioaktiver Zerfall

Die Zerfälle pro Zeiteinheit Δt folgen für $\Delta t \ll t_H$ der Poissonverteilung mit $\lambda = pN$. $p \approx \alpha\Delta t$ ist die Wahrscheinlichkeit für einen Zerfall, N die Anzahl der Teilchen.

Beispiel: $N = 10^{18}$ Atome, $t_H = 4,5$ Mrd. Jahre. Damit $\alpha = \ln 2/t_H$ und für $\Delta t = 1$ s:

$$\lambda = \alpha N \Delta t \approx 4,9.$$

Es gibt also im Mittel 5 Zerfälle pro Sekunde mit der Verteilung:



Mehr **Beispiele**: Übungsblatt

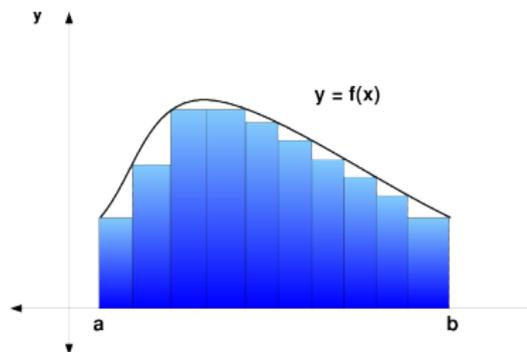
→ Diskrete Verteilungen oft unhandlich. Betrachte **Grenzfälle**:

$$\begin{array}{ccc} \text{Binomial } B_{n,p}(k) & \xrightarrow[n \rightarrow \infty]{\lambda=np=\text{const}} & \text{Poisson } P_\lambda(k) \\ \swarrow n \rightarrow \infty & & \swarrow \lambda \rightarrow \infty \\ & \text{Normal } N_{\mu,\sigma}(x) & \end{array}$$

→ **Kontinuierliche Wahrscheinlichkeitsverteilungen**

Numerische Integration

Um mit kontinuierlichen Wahrscheinlichkeitsverteilungen zu arbeiten, müssen wir nun nicht mehr Summieren, sondern integrieren. Numerisch brauchen wir dafür die **Numerische Integration**.



Dafür sind wichtig:

- ▶ Newton-Cotes-Formeln und zusammengesetzte Formeln
- ▶ Uneigentliche Integrale
- ▶ Gauß-Quadratur

→ siehe Buch Kapitel 9.5

Newton-Cotes Formeln

Nähere Funktion durch **interpolierendes Polynom**:

$$\int_a^b f(x) dx \approx \int_a^b p_N(x) dx$$

$$\int_a^b f(x) dx \approx \int_a^b \sum_{i=0}^N f(x_i) L_i(x) dx = \sum_{i=0}^N f(x_i) \underbrace{\int_a^b L_i(x) dx}_{=:(b-a)w_i} = (b-a) \sum_{i=0}^N w_i f(x_i),$$

Wegen **Runge-Problem** (Ozillationen in Polynomen höherer Ordnung), nutze Newton-Cotes Formeln für **summierte Formeln**:

$$\int_a^b f(x) dx = \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} f(x) dx$$

Beispiel:

Rechteckregel:

$$\int_a^b f(x) dx \approx (b-a)f(x_0)$$

$x_0 = (a+b)/2$: Mittelpunktsregel

Summierte Rechteckregel: $\int_a^b f(x) dx \approx \sum_{j=0}^{n-1} h_j f(x_j)$

Newton-Cotes Formeln

Rechteckregel:

$$\int_a^b f(x) dx \approx (b-a)f(x_0)$$

Trapezregel:

$$\int_a^b f(x) dx \approx (b-a) \sum_{i=0}^1 w_i f(x_i) = \frac{b-a}{2} (f(a) + f(b))$$

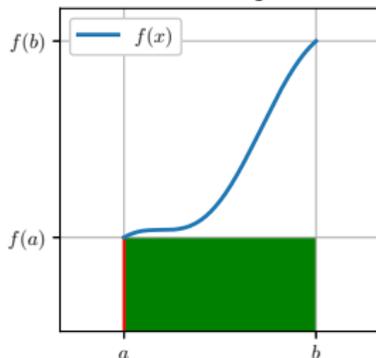
Simpsonregel:

$$\int_a^b f(x) dx \approx (b-a) \sum_{i=0}^2 w_i f(x_i) = \frac{b-a}{6} (f(a) + 4f(\frac{a+b}{2}) + f(b))$$

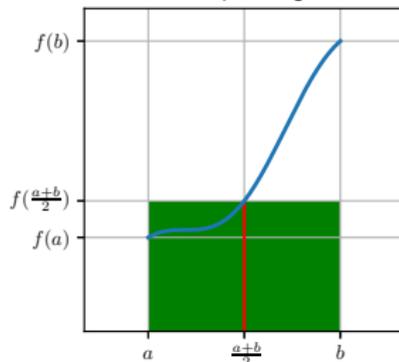
Regel	Stützstellen	Gewichte	Fehler	Sum. Fehler
Rechteck	a	1	$\frac{(b-a)^2}{2} f'(\xi) $	$O(h)$
Mittelpunkt	$\frac{a+b}{2}$	1	$\frac{ b-a ^3}{24} f''(\xi) $	$O(h^2)$
Trapez	a, b	$\frac{1}{2}, \frac{1}{2}$	$\frac{ b-a ^3}{12} f''(\xi) $	$O(h^2)$
Simpson	$a, \frac{a+b}{2}, b$	$\frac{1}{6}, \frac{2}{3}, \frac{1}{6}$	$\frac{ b-a ^5}{2880} f^{(4)}(\xi) $	$O(h^4)$
Simpson-3/8	$a, \frac{a+b}{3}, \frac{2(a+b)}{3}, b$	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	$\frac{ b-a ^5}{6480} f^{(4)}(\xi) $	$O(h^4)$

Newton-Cotes Formeln

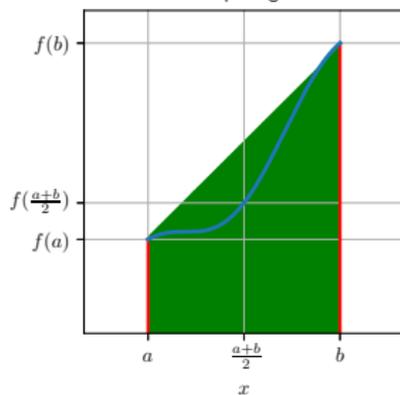
Rechteckregel



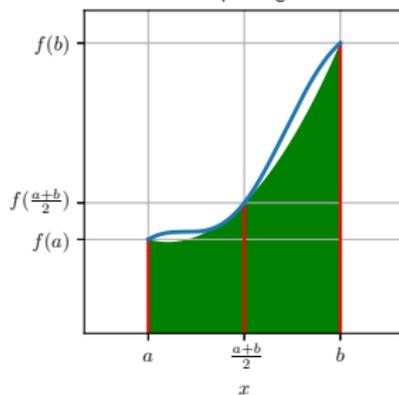
Mittelpunktsregel



Trapezregel



Simpsonregel



Uneigentliche Integrale

Substitution:

$$x(\xi) = \frac{b+a}{2} + \frac{b-a}{2}\xi$$

ändert jedes Integral zu

$$\int_a^b f(x) dx = \frac{b-a}{2} \int_{-1}^1 f(x(\xi)) d\xi.$$

$$\int_a^\infty f(x) dx = \int_0^1 f\left(a + \frac{\xi}{1-\xi}\right) \frac{1}{(1-\xi)^2} d\xi,$$

$$\int_{-\infty}^a f(x) dx = \int_0^1 f\left(a - \frac{1-\xi}{\xi}\right) \frac{1}{\xi^2} d\xi,$$

$$\begin{aligned} \int_{-\infty}^\infty f(x) dx &= \int_{-1}^1 f\left(\frac{\xi}{1-\xi^2}\right) \frac{1+\xi^2}{(1-\xi^2)^2} d\xi \\ &= \int_{-1}^1 \frac{f(\operatorname{atanh} \xi)}{1+\xi^2} d\xi = \int_{-\pi/2}^{\pi/2} \frac{f(\tan \xi)}{\cos^2 \xi} d\xi. \end{aligned}$$

Gauss-Quadratur

Idee: sowohl die Gewichte w_i als auch die Lage der Stützstellen x_i als Unbekannte aufzufassen.

Damit erreicht man, dass mit N Stützstellen Polynome bis zum Grad $2N - 1$ (statt $N - 1$ bei den Newton-Cotes-Formeln) exakt integriert werden.

Beispiel: 2-Punkt-Gauß-Quadratur

$$\int_{-1}^1 (a_0 + a_1x + a_2x^2 + a_3x^3) dx = \sum_{i=0}^1 w_i f(x_i) = w_0 f(x_0) + w_1 f(x_1).$$

$$\begin{aligned} 2a_0 + \frac{2}{3}a_2 &= w_0(a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3) + w_1(a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3) \\ &= a_0(w_0 + w_1) + a_1(w_0x_0 + w_1x_1) + a_2(w_0x_0^2 + w_1x_1^2) \\ &\quad + a_3(w_0x_0^3 + w_1x_1^3). \end{aligned}$$

Koeffizientenvergleich:

$$w_0 + w_1 = 2, w_0x_0 + w_1x_1 = 0, w_0x_0^2 + w_1x_1^2 = \frac{2}{3}, w_0x_0^3 + w_1x_1^3 = 0,$$

mit der einfach zu prüfenden Lösung $w_0 = w_1 = 1$, $x_0 = 1/\sqrt{3}$, $x_1 = -1/\sqrt{3}$.

Wir können daher mit nur zwei Stützstellen jedes Polynom bis zum Grad 3 exakt integrieren. Die 2-Punkt-Gauß-Quadratur lautet

$$\boxed{\int_{-1}^1 f(x) dx \approx f\left(\frac{1}{\sqrt{3}}\right) + f\left(-\frac{1}{\sqrt{3}}\right)}. \quad (1)$$

Gauss-Quadratur

Für mehr als zwei Stützstellen: x_i sind Nullstellen der Legendre-Polynome mit $w_i = \int_{-1}^1 L_i(x) dx$.

Also, z. B. 3-Punkt-Gauß-Legendre-Formel

$$\int_{-1}^1 f(x) dx \approx \frac{5}{9} f\left(\sqrt{\frac{3}{5}}\right) + \frac{8}{9} f(0) + \frac{5}{9} f\left(-\sqrt{\frac{3}{5}}\right), \quad (2)$$

die Polynome bis zum Grad 5 exakt integriert.

Tabelle: Übersicht über die verallgemeinerten Gauß-Quadratur-Methoden

Name	Intervall	Gewichtsfunktion	Orthogonale Polynome
Gauß-Legendre	$[-1, 1]$	$g(x)$ 1	Legendre-Polynome
Gauß-Laguerre	$[0, \infty]$	e^{-x}	Laguerre-Polynome
Gauß-Hermite	$[-\infty, \infty]$	e^{-x^2}	Hermite-Polynome
Gauß-Tschebyschow	$[-1, 1]$	$1/\sqrt{1-x^2}$	Tschebyschow-Polynome

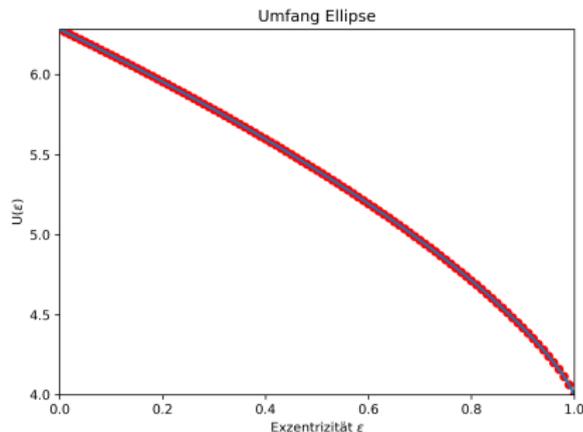
Num. Integration - Beispiel: Kurvenlänge einer Ellipse

Mit $\gamma(t) = \begin{pmatrix} a \cos t \\ b \sin t \end{pmatrix}$ ($t \in [0, 2\pi]$):

$$|\gamma(t)| = \sqrt{a^2 \sin^2 t + b^2 \cos^2 t} = a\sqrt{1 - \varepsilon^2 \cos^2 t}.$$

Damit ist die Kurvenlänge:

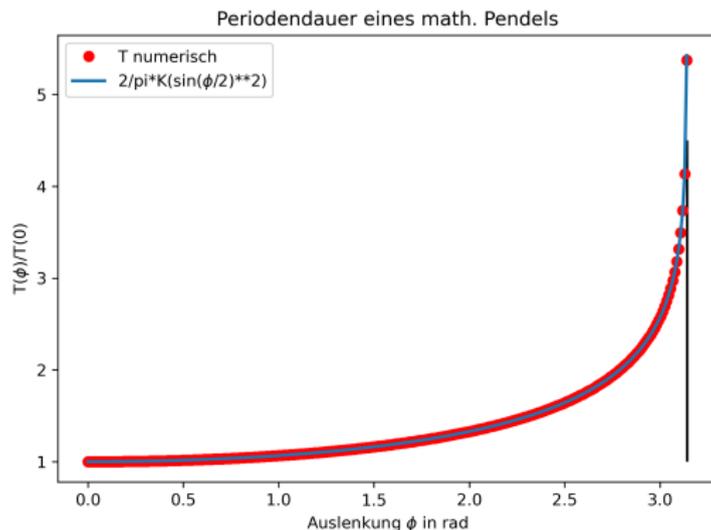
$$L = \int_0^{2\pi} |\gamma(t)| dt = 4a \underbrace{\int_0^{\pi/2} \sqrt{1 - \varepsilon^2 \cos^2 t} dt}_{E(\varepsilon^2), \text{Elliptisches Integral}}.$$



Num. Integration - Beispiel: Periodendauer des math. Pendels

$$T = \frac{4}{\omega} \int_0^{\pi/2} \frac{1}{\sqrt{1 - \sin^2 \frac{\varphi_0}{2} \sin^2 \varphi}} d\varphi = \frac{4}{\omega} K(\sin^2 \frac{\varphi_0}{2}).$$

`scipy.special.elliptic(k**2)`



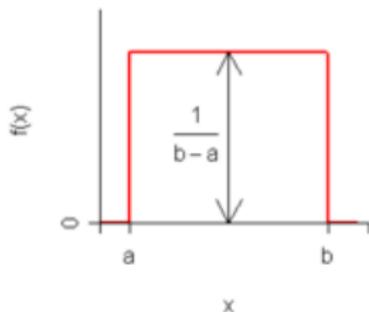
- ▶ Gleichverteilung
- ▶ Exponentialverteilung
- ▶ Lorentz-Verteilung
- ▶ Normalverteilung

siehe auch Maxwell-Boltzmann (Buch)

Mehr Beispiele: Übung

$$P(x) = \frac{1}{b-a} \quad (a \leq x \leq b)$$

$X \sim \text{Gleich}(a, b)$



- ▶ Normierung: $\int_{-\infty}^{\infty} P(x) dx = \int_a^b dx = \dots = 1$
- ▶ Erwartungswert: $\mu = \int_a^b xP(x) dx = \dots = \frac{a+b}{2}$
- ▶ Varianz: $\sigma^2 = \int_a^b (x - \mu)^2 P(x) dx = \dots = \frac{(b-a)^2}{12}$
- ▶ \rightarrow Standardabweichung für $[0, 1]$: $\sigma_{[0,1]} = 1/\sqrt{12}$

Exponentialverteilung

Betrachtet man Ereignisse, deren Anzahl in einem beliebigen Zeitintervall Poisson-verteilt ist (sog. Poisson-Prozesse), und fragt sich, wie der zeitliche Abstand einzelner Ereignisse verteilt ist, so gelangt man zur Exponentialverteilung. Diese wird für $t \geq 0$ mit der Funktion

$$p_\alpha(t) = \alpha e^{-\alpha t} \quad (3)$$

Normierung: $\int_0^\infty p_\alpha(t) dt = 1$

Erwartungswert:

$$\mu = \int_0^\infty t p_\alpha(t) dt = \frac{1}{\alpha}. \quad (4)$$

Man erwartet also eine mittlere Lebensdauer von $\tau = 1/\alpha$ mit $p(\tau) = \alpha/e$.

Kummulierte Verteilung:

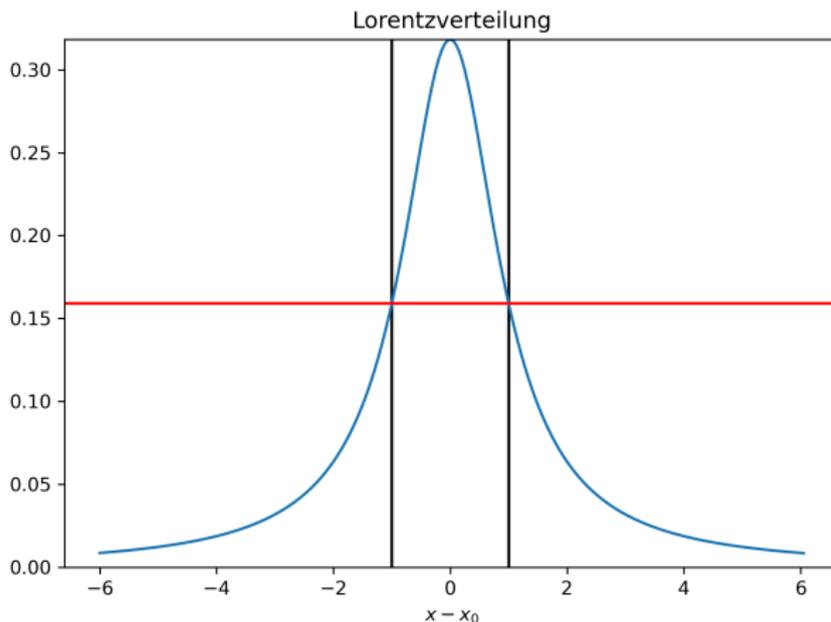
$$P_\alpha(t) = \int_0^t p_\alpha(t') dt' = \alpha \int_0^t e^{-\alpha t'} dt' = 1 - e^{-\alpha t}. \quad (5)$$

Für $t = \tau = 1/\alpha$ ist die Verteilung auf $p(t = \tau) = \alpha/e$ abgefallen und wegen $P(t = \tau) = 1 - 1/e \approx 0,63$ damit ca. 63 % zerfallen. Genau die Hälfte ist zerfallen, wenn $P_\alpha(t_H) = 1/2$. Damit folgt $t_H = \ln(2)/\alpha$ und $p(t_H) = \alpha/2$, also ist genau die Hälfte zerfallen, wenn die Verteilung auf die Hälfte abgefallen ist.

Lorentz-Verteilung

(In der Mathematik: Cauchy-Lorentz-Verteilung)

$$f(x) = \frac{1}{\pi} \frac{\gamma}{\gamma^2 + (x - x_0)^2}$$



Ursprung: Fourier-Trafo vom exp. Zerfall

Daher bekannt aus: Resonanzkurve, Spektrallinie,

"Breit-Wigner"-Resonanz

Eigenschaften:

- ▶ Erwartungswert: $\mu = \int_{-\infty}^{\infty} xf(x) dx = \infty!$
- ▶ Varianz: $\sigma^2 = \int_{-\infty}^{\infty} x^2 f(x) dx = \infty!$

Verwende daher Median x_0 und FWHM (*full width half maximum*)

$\Gamma = 2\gamma$ als Linienbreite.

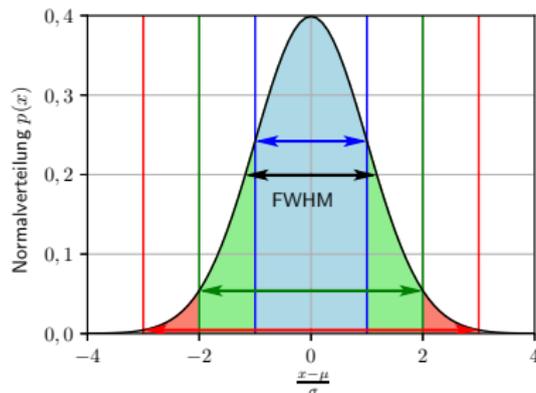
Vertrauensintervall: $\int_{-\Gamma/2}^{\Gamma/2} f(x) dx = 50\%$.

CDF: $F(x) = \int_{-\infty}^x f(x') dx' = \frac{1}{\pi} \operatorname{atan}\left(\frac{x-x_0}{\gamma}\right) + \frac{1}{2}$.

Normalverteilung

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$e^{-\frac{x_H^2}{2\sigma^2}} = \frac{1}{2} \Rightarrow \text{FWHM} = 2x_H = 2\sqrt{2\ln 2}\sigma \approx 2,3548\sigma.$$



$$P(x) = P(\leq x) = \frac{1}{2} \left(1 + \operatorname{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)\right), \quad \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

Suche anhand einer Stichprobe die **besten Parameter** einer Verteilung bzw. Funktion

→ **Anpassung** der gesuchten Verteilung/Funktion (Modell) an die Stichprobe/Messwerte durch Optimierung der Parameter

2 wichtige **Verfahren**:

- ▶ **Kleinste-Quadrate-Methode** (*least square*, LS) Kap. 9.6
- ▶ **Maximum Likelihood-Methode** [ML] → Kap. 18.2.1



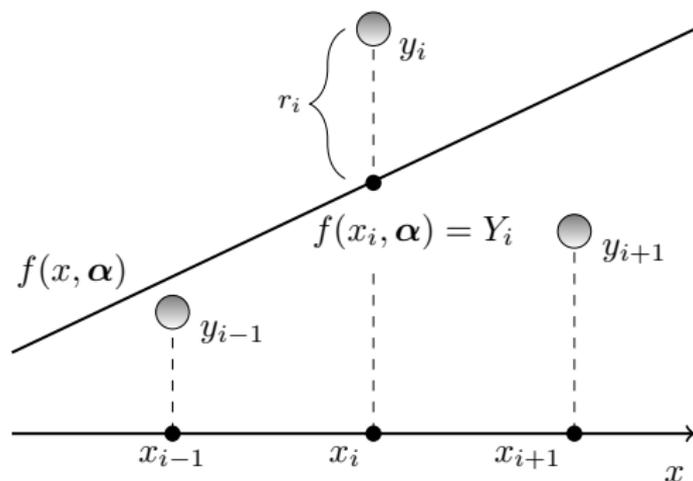
Anpassung an Daten

Anpassung (*fit*) einer Modellfunktion f an (Mess-)Daten durch **Optimierung der Modellparameter**.

Messwerte: $y_i(x_i)$ ($i = 1, \dots, n$)

Modellfunktion: $f(x; \underbrace{\alpha_1, \dots, \alpha_m}_{=\alpha, \text{Modellparameter}})$

→ Suche Parameter α , sodass die Abweichungen (Residuen) des Modell von den Daten möglichst klein sind.



Kleinste-Quadrate-Methode

→ Minimiere die quadratische Abweichungen der Modellfunktion (f) von den Daten (y):

$$\min_{\alpha} \sum_{i=1}^N (y_i - f(x_i, \alpha))^2 = \min_{\alpha} \sum_{i=1}^N r_i^2(\alpha) = \min_{\alpha} \chi^2(\alpha). \quad (6)$$

χ^2 : sum of squared residuals (SSR) / residual sum of squares (RSS)

Beispiele:

- ▶ **Lineare Anpassung**/Regression: Modellfunktion ist linear ($f(x; a, b) = a + bx$) →. Optimale Parameter lassen sich direkt berechnen. (→ Kapitel 9.6.1)
- ▶ **Polynomregression**: Modellfunktion ist ein Polynom (→ Kapitel 9.6.2)
- ▶ **Polynomapproximation**: Anpassung einer bel. Funktion mit einem Polynom (→ Kapitel 18.2.2)
- ▶ Allgemeiner Fall: **Nichtlineare Anpassung**

Lineare Regression

Modellfunktion: $f(x; a, b) = a + bx$, $\langle x \rangle = \frac{1}{n} \sum_i x_i$

$$\min_{\alpha} \chi^2(\alpha) = \min_{a,b} \sum_{i=1}^n (y_i - a - bx_i)^2 :$$

$$0 = \frac{d\chi^2}{db} = -2 \sum_i (y_i - a - bx_i)x_i$$

$$\sum_i x_i y_i = b \sum_i x_i^2 + a \sum_i x_i$$

$$\langle xy \rangle = b \langle x^2 \rangle + a \langle x \rangle \quad (7)$$

$$0 = \frac{d\chi^2}{da} = -2 \sum_i (y_i - a - bx_i)$$

$$\sum_i y_i = b \sum_i x_i + a \sum_i 1$$

$$\langle y \rangle = b \langle x \rangle + a \quad (8)$$

$$\begin{pmatrix} \langle y \rangle \\ \langle xy \rangle \end{pmatrix} = \begin{pmatrix} 1 & \langle x \rangle \\ \langle x \rangle & \langle x^2 \rangle \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \quad (9)$$

$$a = \langle y \rangle - b \langle x \rangle \quad (10)$$

$$b = \frac{\langle xy \rangle - \langle x \rangle \langle y \rangle}{\langle x^2 \rangle - \langle x \rangle^2} = \frac{\text{cov}(x, y)}{\sigma_x^2} \quad (11)$$

Ergebnis:

$$\begin{pmatrix} b \\ a \end{pmatrix} = \frac{1}{\sigma_x^2} \begin{pmatrix} \text{cov}(x, y) \\ \langle x^2 \rangle \langle y \rangle - \langle x \rangle \langle xy \rangle \end{pmatrix} \quad (12)$$

Kovarianz:

$$\text{cov}(x, y) = \langle (x - \langle x \rangle)(y - \langle y \rangle) \rangle = \langle xy + \langle x \rangle \langle y \rangle - x \langle x \rangle - y \langle x \rangle \rangle = \langle xy \rangle - \langle x \rangle \langle y \rangle$$

$$\text{cov}(x, y) = 0 \rightarrow \langle xy \rangle = \langle x \rangle \langle y \rangle \text{ (unkorreliert)}$$

$$\text{cov}(x, x) = \langle (x - \langle x \rangle)^2 \rangle = \langle x^2 \rangle - \langle x \rangle^2 = \sigma_x^2$$

Kovarianzmatrix:

$$C(x, y) = \begin{pmatrix} \text{cov}(x, x) & \text{cov}(x, y) \\ \text{cov}(y, x) & \text{cov}(y, y) \end{pmatrix} = \begin{pmatrix} \sigma_x^2 & \text{cov}(x, y) \\ \text{cov}(x, y) & \sigma_y^2 \end{pmatrix} \quad (13)$$

Lineare Regression

Korrelationskoeff. (norm. Kovarianz):

$$\rho_{x,y} = \frac{\text{cov}(x,y)}{\sqrt{\sigma_x^2 \sigma_y^2}}$$

→ Correlation.png

Beispiel:

x_i	1	2	3
y_i	1	3	2

$$\langle x \rangle = 2, \langle y \rangle = 2, \langle xy \rangle = 13/3, \langle x^2 \rangle = \langle y^2 \rangle = 14/3$$

$$\rightarrow \sigma_x^2 = \langle x^2 \rangle - \langle x \rangle^2 = 2/3, \sigma_y^2 = \langle y^2 \rangle - \langle y \rangle^2 = 2/3$$

$$\text{cov}(x,y) = \langle xy \rangle - \langle x \rangle \langle y \rangle = 1/3$$

$$b = 1/3 / 2/3 = 1/2$$

$$a = 2 - 1/2 * 2 = 1$$

$f(x) = x/2 + 1$

$$\rho_{x,y} = \frac{1/3}{\sqrt{2/3 \cdot 2/3}} = 1/2 > 0$$

Über die genaue Bezeichnung und ihre Abkürzungen gibt es international keine Einigkeit. Die natürliche deutsche Abkürzung für die Residuenquadratsumme bzw. die Summe der (Abweichungs-)Quadrate der Restabweichungen (oder: "Residuen"), ist SAQRest, oder SQR. Die englische Abkürzung SSR ist vieldeutig und führt zu anhaltenden Verwechslungen: Sowohl Sum of Squared Residuals (Residuenquadratsumme) als auch Sum of Squares due to Regression (Regressionsquadratsumme) werden als SSR abgekürzt. Allerdings wird die Regressionsquadratsumme oft auch als erklärte Quadratsumme (Sum of Squares Explained) bezeichnet, deren natürliche englische Abkürzung SSE ist. Die Abkürzungsproblematik wird dadurch verschärft, dass die Residuenquadratsumme oft auch als Fehlerquadratsumme (Sum of Squares Error) bezeichnet wird, deren natürliche englische Abkürzung ebenfalls SSE ist (diese Bezeichnung ist besonders irreführend, da die Fehler und die Residuen unterschiedliche Größen sind). Des Weiteren findet sich für Residuenquadratsumme ebenfalls die englische Abkürzung RSS, statt der Abkürzung SSR, da statt der Bezeichnung Sum of Squared Residuals, oft auch die Bezeichnung Residual Sum of Squares verwendet wird. Auch diese englische Abkürzung kann mit der Regressionsquadratsumme verwechselt werden, die im Englischen auch als Regression Sum of Squares bezeichnet, deren natürliche englische Abkürzung auch hier RSS ist.
(<https://de.wikipedia.org/wiki/Residuenquadratsumme>)

Lineare Regression

- ▶ $\sum_i (y_i - \langle y \rangle)^2 = \sigma_y^2$: sum of squared total (**SST**) → "gesamte" Variation
- ▶ $\sum_i (f(x_i) - \langle y \rangle)^2 = \sigma_{\hat{y}}^2$: sum of squares explained (**SSE**), *sum of squares due to regression (SSR), regression sum of squares (RSS)* → "vorhergesagte" Variation
- ▶ $\sum_i (f(x_i) - y_i)^2$: sum of squared residuals (**SSR**), *residual sum of squares (RSS), sum of squared errors (SSE)* → "restliche" Variation

$$SST = SSE + SSR$$

Güte der Anpassung: $R^2 = \frac{SSE}{SST} \rightarrow 1$ (Wieviel der Variationen wird durch das Model erklärt)

Güte der Anpassung: $\chi^2 \rightarrow 0$ (Wieviel der Variationen bleibt übrig)

Unsicherheit:

$MSE = SSR / (n - p) = SST(1 - R^2) / (n - p)$, p - Anzahl Parameter

$\Delta b = \sqrt{MSE} / \sigma_x$

→ linreg.py

NumPy:

`x.mean()`, `x.var()`, `numpy.cov(x,y)`

bias - Normierung mit N, statt N-1

→ `linreg-numpy.py`

SciPy:

`scipy.stats.linregress()`

- ▶ intercept: a
- ▶ slope: b
- ▶ rvalue: $\sqrt{R^2}$
- ▶ pvalue: Hypothesentest auf $b=0$
- ▶ stderr: σ_b
- ▶ intercept_stderr: σ_a (ab SciPy 1.6)

→ `linreg-scipy.py`

Regression mit Unsicherheiten

Berücksichtigung von Unsicherheiten:

$$y_i \pm \sigma_i$$

$$\chi^2 = \sum_i w_i (y_i - f(x_i, \alpha))^2$$

Gewichtung: $w_i = 1/\sigma_i^2$

Ergebnis (WSSR):

$$\chi(a, b)^2 = \sum_i \left(\frac{y_i - a - bx_i}{\sigma_i} \right)^2$$

$$\begin{pmatrix} \langle y \rangle \\ \langle xy \rangle \end{pmatrix} = \begin{pmatrix} 1 & \langle x \rangle \\ \langle x \rangle & \langle x^2 \rangle \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

mit $\langle x \rangle = \sum_i w_i x_i$, etc. (vgl. Mittelwertbildung von diskreten Verteilungen)

- ▶ Erweiterung: **Anpassung eines Polynoms** an Daten:
Buch Kap. 9.6.2
numpy.polyfit
polyreg-numpy.py
polyreg-numpy-interpolation.py
- ▶ Anwendung Kleinste-Quadrate: (globale) **Näherung einer Funktion** durch ein Polynom:
Buch Kap. 18.2.2
polyfit.py
polyfit-legendre.py

Lösung des Optimierungsproblems für eine allg. Modellfunktion mithilfe von Minimierungsmethoden der Linearen Algebra (→ Kapitel 18.2.3)

Idee:

Minimierung durch Anwendung des (mehrdimensionalen) Newton-Verfahrens auf den Gradienten der Modellfunktion. Bei Verwendung von $\chi^2(\alpha)$ ergibt sich das **Gauß-Newton-Verfahren**. Um die Konvergenz zu garantieren, führt man einen Parameter λ ein und erhält endlich den **Levenberg-Marquardt-Algorithmus**. (→ Kapitel 11.3)

Nichtlineare Anpassung

(Iteratives) Finden von Nullstellen mit dem Newton-Verfahren:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Mehrdimensionales **Newton-Verfahren**:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - J_f^{-1}(\mathbf{x}_n)\mathbf{f}(\mathbf{x}_n)$$

$J_f = (\partial_{x_j} f_i)$ - Jacobi-Matrix
bzw. löse LGS

$$J(\mathbf{x}_n) \underbrace{(\mathbf{x}_{n+1} - \mathbf{x}_n)}_{\Delta \mathbf{x}_n} = -\mathbf{f}(\mathbf{x}_n)$$

und berechne $\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta \mathbf{x}_n$.



Nichtlineare Anpassung

Optimierungsaufgabe: Finde Nullstellen der Ableitung von $f(\mathbf{x})$

$$\mathbf{x}_{n+1} = \mathbf{x}_n - J_{\nabla f}^{-1}(\mathbf{x}_n) \nabla f(\mathbf{x}_n)$$

mit $J_{\nabla f} = H_f = \left(\frac{\partial^2 f}{\partial x_i \partial x_j} \right)$ - Hesse-Matrix

bzw.

$$H_f \Delta \mathbf{x}_n = -\nabla f(\mathbf{x}_n)$$

Wende **Newton-Verfahren** auf **Methode der kleinsten Quadrate** an:
Suche **Minimum** von $\chi^2(\boldsymbol{\alpha})$:

$$\mathbf{0} = \nabla_{\boldsymbol{\alpha}} \chi^2(\boldsymbol{\alpha}) = 2J_r^T(\boldsymbol{\alpha}) \cdot \mathbf{r}(\boldsymbol{\alpha})$$

In Newton-Verfahren:

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n - J_{\nabla \chi^2}^{-1}(\boldsymbol{\alpha}_n) \cdot \nabla \chi^2(\boldsymbol{\alpha}_n)$$

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n - (J_r^T \cdot J_r)^{-1} \cdot J_r^T(\boldsymbol{\alpha}_n) \cdot \mathbf{r}(\boldsymbol{\alpha}_n)$$

bzw.

$$(J_r^T \cdot J_r) \Delta \boldsymbol{\alpha}_n = -J_r^T(\boldsymbol{\alpha}_n) \cdot \mathbf{r}(\boldsymbol{\alpha}_n)$$

genannt **Gauss-Newton**

Nichtlineare Anpassung

Gauss-Newton Verfahren: Keine garantierte Konvergenz (vgl. Newton-Verfahren)



Lösung: erzwinge Abstieg durch **Marquardt-Parameter** λ_n :

$$(J_r^T \cdot J_r + \lambda_n D) \Delta \alpha_n = -J_r^T(\alpha_n) \cdot r(\alpha_n)$$

$\lambda \rightarrow 0$: Gauss-Newton

$\lambda \rightarrow \infty$: $\alpha_{n+1} = \alpha_n - \gamma_n \nabla \chi^2(\alpha_n)$ Gradientensuchverfahren

$D = \text{diag}(J^T \cdot J)$ (Levenberg):

$$(J^T \cdot J + \lambda_n \text{diag}(J^T \cdot J)) \Delta \alpha_n = -J^T(\alpha_n) \cdot r(\alpha_n)$$

genannt **Levenberg-Marquardt-Algorithmus**

Nichtlineare Anpassung - Python

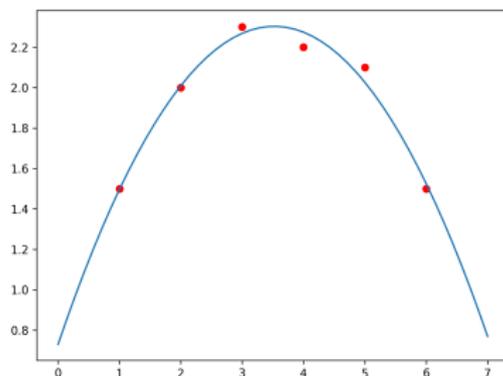
```
popt, pcov = scipy.optimize.curve_fit(f,x,y)
```

popt - opt. Parameter

pcov - Kovarianzmatrix der Parameter

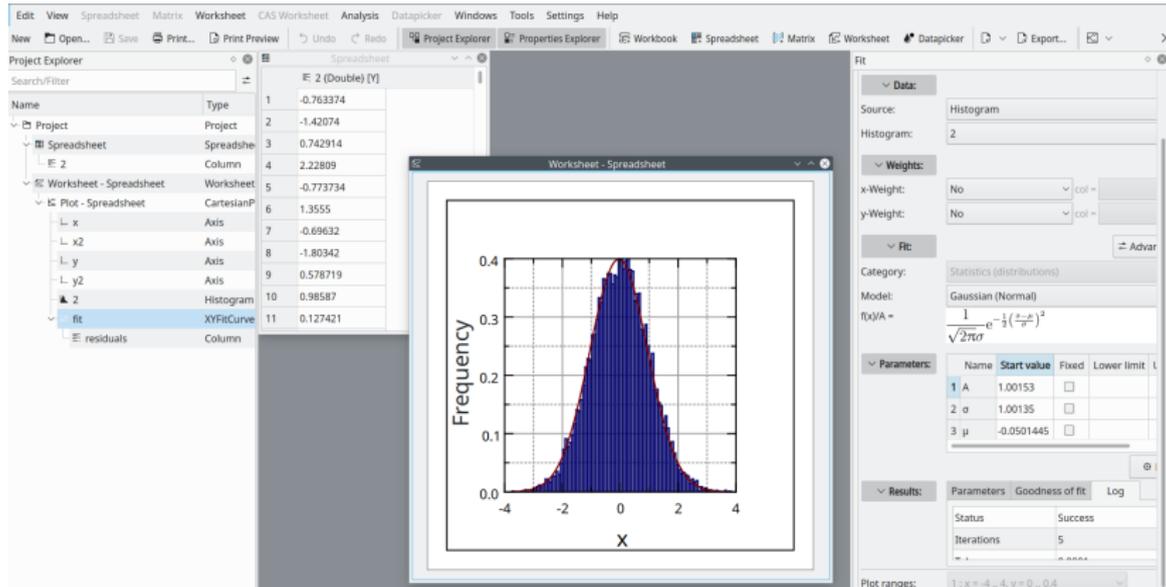
→ `nonlin-scipy-curve_fit.py`

```
1 Parameter :  
2 [ 0.73          0.89321429 -0.12678571]  
3 Kovarianz :  
4 [[ 0.01348572 -0.00821786  0.00105357]  
5 [-0.00821786  0.00577207 -0.00079018]  
6 [ 0.00105357 -0.00079018  0.00011288]]
```



Nichtlineare Anpassung - Programme

GUI-Programme mit Fitfunktionen: LabPlot, Origin, Igorpro, ...



Maximum-Likelihood-Methode

Suche **Parameter** α einer vermuteten/bekanntes **Verteilung** $f(y; \alpha)$ anhand einer Stichprobe y_i (n unabhängige Messungen).

Maximum-Likelihood: Die besten Parameter ergeben die **höchste Wahrscheinlichkeit** die Stichprobe zu erhalten.

Likelihood-Funktion: Datenpunkte unabhängig, d.h. Wahrscheinlichkeiten multiplizieren sich:

$$P = f(y_1, y_2, \dots, y_n; \alpha) = f(y_1; \alpha) \cdots f(y_n; \alpha) = \prod_{i=1}^n f(y_i; \alpha) =: L(\alpha)$$

Lösung:

$$\left. \frac{\partial L(\alpha)}{\partial \alpha} \right|_{\hat{\alpha}} = \mathbf{0} \quad \text{bzw.} \quad \left. \frac{\partial \ln L(\alpha)}{\partial \alpha} \right|_{\hat{\alpha}} = \mathbf{0}$$

(**Log-Likelihood-Funktion** meist einfacher zu berechnen, da Logarithmus von Produkt = Summe)

Maximum-Likelihood-Methode - Beispiel (Fische im Bodensee)

Stichprobe: Einen Tag angeln. Ergebnis: 14 Felchen, 6 Saiblinge.

Likelihood-Funktion (Binomialverteilung):

$$L(p) = B_{n,k}(p) = \binom{n}{k} p^k (1-p)^{n-k}$$

$$\ln L(p) = \ln \binom{n}{k} + k \ln p + (n-k) \ln(1-p)$$

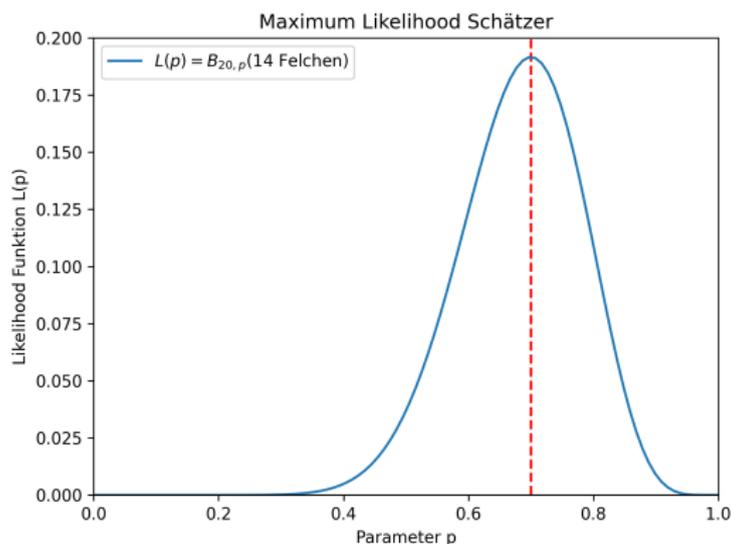
$$\frac{d}{dp} \ln L(p) \Big|_{\hat{p}} = \frac{x}{\hat{p}} - \frac{n-k}{1-\hat{p}} \stackrel{!}{=} 0$$

$$\rightarrow \boxed{\hat{p} = \frac{k}{n}}$$

Maximum-Likelihood-Methode - Beispiel (Fische im Bodensee)

Stichprobe: Einen Tag angeln. Ergebnis: 14 Felchen, 6 Saiblinge.

$$\hat{p} = \frac{k}{n} = 14/21 = 70\%$$



$$f(k; \lambda) = \frac{\lambda^k}{k!} e^{-\lambda}$$

Stichprobe k_1, \dots, k_n . Likelihood-Funktion:

$$L(\lambda) = \frac{\lambda^{\sum_i k_i}}{\prod_i k_i!} e^{-n\lambda}$$

$$\ln L(\lambda) = \sum_i (k_i \ln \lambda - \ln(k_i!) - \lambda)$$

$$\frac{d}{d\lambda} \ln L(\lambda) \Big|_{\hat{\lambda}} \stackrel{!}{=} 0$$

$$\hat{\lambda} = \frac{1}{n} \sum_i k_i = \langle k \rangle$$

$$f(x; \lambda) = \lambda e^{-\lambda x} \quad (\lambda > 0)$$

Stichprobe x_1, \dots, x_n . Likelihood-Funktion:

$$L(\lambda) = \lambda^n e^{-\lambda \sum_i x_i}$$

$$\ln L(\lambda) = n \ln \lambda - \lambda \sum_i x_i$$

$$\frac{d}{d\lambda} \ln L(\lambda) \Big|_{\hat{\lambda}} \stackrel{!}{=} 0$$

$$\hat{\lambda} = \frac{1}{\frac{1}{n} \sum_i x_i} = \frac{1}{\langle x \rangle}$$

$$f(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Stichprobe x_1, \dots, x_n . Likelihood-Funktion:

$$L(\mu, \sigma^2) = \frac{1}{(\sigma\sqrt{2\pi})^n} e^{-\frac{1}{2\sigma^2} \sum_i (x_i - \mu)^2}$$

$$\ln L(\mu, \sigma^2) = -\frac{n}{2}(\ln(2\pi) - \ln \sigma^2) - \frac{1}{2\sigma^2} \sum_i (x_i - \mu)^2$$

$$\frac{\partial}{\partial \mu} \ln L(\mu, \sigma^2) \Big|_{\hat{\mu}} = -\frac{1}{\sigma^2} \sum_i (x_i - \hat{\mu}) \stackrel{!}{=} 0$$

$$\hat{\mu} = \frac{1}{n} \sum_i x_i = \langle x \rangle.$$

analog $\left(\frac{\partial}{\partial \sigma^2}\right)$:

$$\hat{\sigma}^2 = \frac{1}{n} \sum_i (x_i - \mu)^2.$$