Übung 4: Regressionskurven

22. Mai 2017

https://cloud.ksz.ch

Ueb4_....i

Erste Ideen für Aufgabe 1

- Umfang berechnen (Modell Kreis) $2 \cdot \pi \cdot r$
- np.polyfit(...) Polynomfunktion einpassen
- sp.optimite → curve_fit(...) Beliebige Funktion anpassen.

Mögliche Modelle:

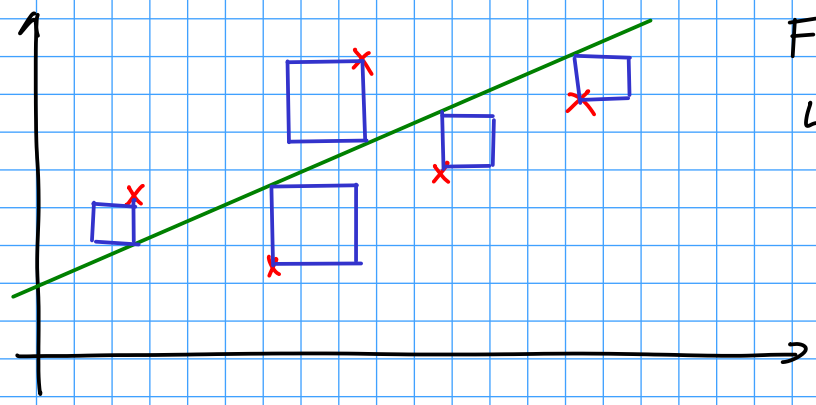
- Quadr. Fkt: $f(x) = ax^2 + bx + c$
- Exponential: $f(x) = a \cdot b^x$ ← ? $f(1) = a$
- Polynomfunktion n-ten Grades: $a_n \cdot x^n + a_{n-1} x^{n-1} + \cdots + a_0$ ←
- Potenzfunktion: $f(x) = a \cdot x^p$ ←
- Linear: $f(x) = mx + q$

① Pysik: 3. Keplersche Satz

② Lineare Regression

③ curve_fit()

Mit lineare Regression



FQS soll minimal werden.

Beispiel: 1. $f(x) = x^2$ $\quad y = x^2$ $\quad | \log()$

$\log(y) = 2\log(x)$ $\quad$ Subst. $\hat{y} := \log(y)$

$\hat{y} = 2\log(x)$ $\quad$ Subst. $\hat{x} := \log(x)$

$\hat{y} = 2\hat{x}$

2. $f(x) = 2^x$ $\quad | \log()$

$\log(y) = x \cdot \log(2)$ $\quad$ Subst: $\hat{y} := \log(y)$

$\hat{y} = \log(2) \cdot x$

Modelle: x und y logarithmiert: $y = a \cdot x^p$

y logarithmiert: $y = a \cdot b^x$

Aufgabe 1

a) $y = a \cdot x^r$ $\quad | \log()$

$\log(y) = \log(a) + p \cdot \log(x) \Rightarrow \hat{y} = \underbrace{\log(a)}_{q} + \underbrace{p}_{m} \cdot \hat{x}$

$m = p$

$q = \log(a)$

q: y-Achsenabschnitt

m: Steigung

$a = e^q = \exp(q)$

Die Kurve ist: $f(x) = 0.0005469 \cdot x^{1.4999}$

```
#plt.plot(a, T, 'o', color='b')
log_a = np.log(a)
log_T = np.log(T)
m,q = sp.stats.linregress(log_a, log_T)[:2]
p = m
A = np.exp(q)
print(A,p)
def f(x):
    return A*x**p
x = np.linspace(0,5900)
plt.plot(a,T, 'o')
plt.plot(x, f(x))
```

Python-Funktion: def, lambda

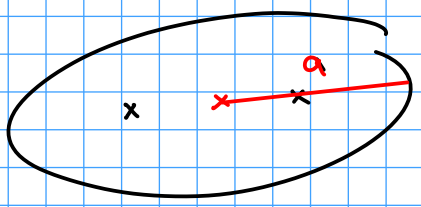
29. Mai 2017

① - Lösung mit physikalischen Formeln
- Lösung mit curve_fit

① Pysikalisches Gesetz

T: Umlaufzeit
a: grosse Halbachse

$$\frac{T^2}{a^3} = k \quad \text{(Konstante } k\text{)}$$



Funktion $a \longrightarrow T$

$$T^2 = a^3 \cdot k$$

$$T(a) = \sqrt{a^3 \cdot k} = (a^3 \cdot k)^{\frac{1}{2}} = a\,(a \cdot k)^{\frac{1}{2}} = a^{\frac{3}{2}} \cdot k^{\frac{1}{2}}$$

Was ist k?
Durchschnitt des Verhältnisses aller Planeten.

Logarithmieren in y-Richtung

$$f(x) = a \cdot b^x \overset{\log}{\Rightarrow} \log(y) = \log(a) + x \cdot \log(b)$$

$$\hat{y} = mx + q$$

Wobei $m = \log(b) \Rightarrow b = e^m$

$q = \log(a) \Rightarrow a = e^q$

in Python $\log = \ln$. $e^x = np.exp(x)$

x-Achse logarithmieren

~~$y = a \cdot \log_b(x)$~~ $\Rightarrow$ $\hat{x} = \log_e(x)$

$$y = a \cdot \frac{\log_e(x)}{\log_e(b)} = \frac{a}{\log_e(b)} \log(x)$$

Einfacher Fall: $b = e$ oder $a = 1$

$$y = a \cdot \ln(x) + b$$

Beide Achsen logarithmiert

$f(x) = a \cdot x^p \quad | \log$

$\log(y) = \log(a) + p \cdot \log(x)$

$\hat{x} = \log(x) \quad \hat{y} = \log(\hat{y})$

$\hat{y} = p \cdot x + \log(a)$

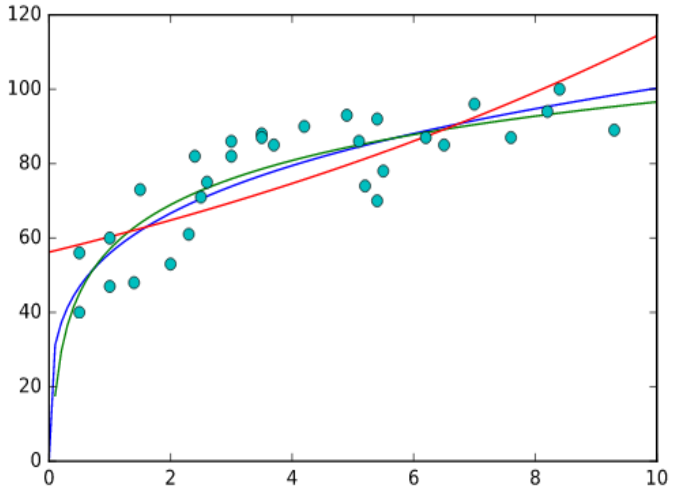
d.h. $m = p$

$q = \log(a)$

---

6. Juni 2017

Aufgabe 2



m: Exponentialfunktion

m: Potenzfunktion } etwa gleich gut.

m: Logarithmus

- $f(x) = a \cdot x^p$ mit $0 < p < 1$
- $f(x) = a \cdot \ln(x) + b$

↳ Haben keinen Grenzwert.

Beispiel: $f(x) = \sqrt[3]{x} = x^{\frac{1}{3}}$ $\quad \lim_{x \to \infty} x^{\frac{1}{3}} = n$

$x^{\frac{1}{3}} \leq n$

$x \leq n^3$ ↯

$\Rightarrow$ Es gibt keinen Grenzwert.

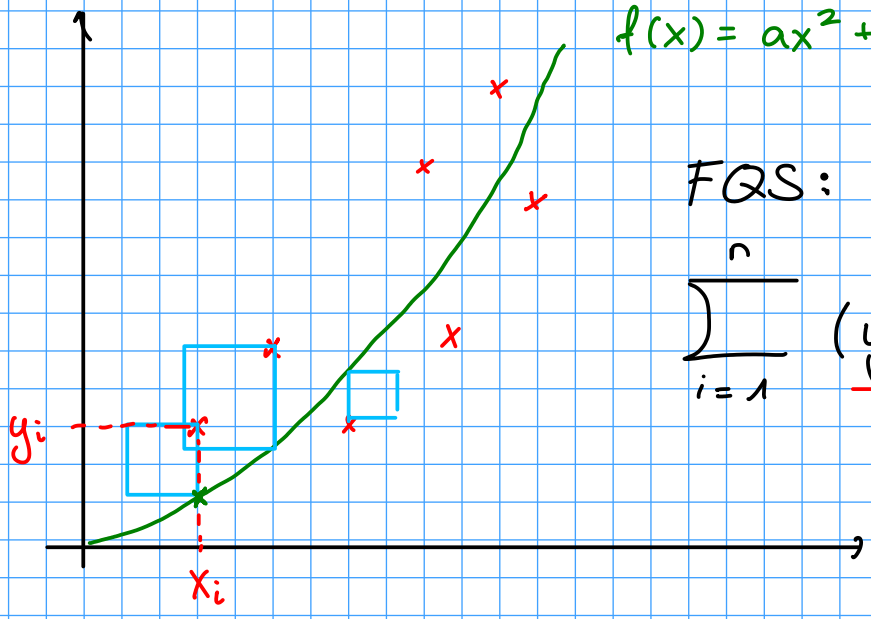
Aufgabe 3

Körpergrösse:	187	173	177	175	184	174	183	175
Spannweite:	187	174	178	176	180	171	191	171
Fusslänge:	26	26	26	27.5	25	28	32	25

Alle Angaben in cm.

Körpergrösse → Spannweite: Potenzfunktion $f(x) = a \cdot x^p$

Körpergrösse → Fusslänge: Alle Modelle schlecht, Exponential?



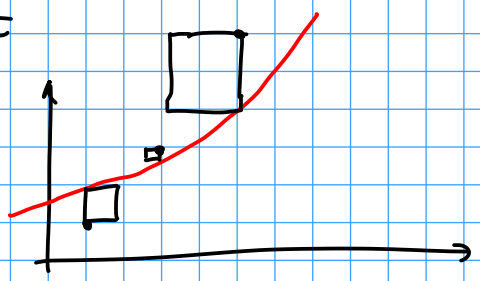
FQS:

$$\sum_{i=1}^{n} (y_i - f(x_i)) = FQS(a,b,c)$$

from scipy.optimize import minimize

Übung 5

- Linearisieren → FQS minimiert
- FQS direkt berechnen.



$$f(x) = a \cdot b^x$$

Repetition: (x_i) (y_i) $f(x)$

FQS: $\sum_{i=1}^{n} (y_i - f(x_i))^2$

```
xs = [2,4,5,7,9]
ys = [9.2, 49.8, 117.9, 613.6, 3226.5]

def exp_mod(x, a, b):
    return a*b**x

vals = curve_fit(exp_mod, xs, ys)
a, b = vals[0]
xlin = np.linspace(1,10)
plt.plot(xlin, exp_mod(xlin, a, b))
```

Relatives Fehler: Fehler relativ zum Messwert

Relative Fehler: $\frac{y_i - f(x_i)}{y_i}$

```
def fqs(f, xs, ys):
    fqs = 0
    for punkt in zip(xs, ys):
        x = punkt[0]
        y = punkt[1]
        fqs += (y - f(x))**2
    return fqs

vals = curve_fit(exp_mod, xs, ys)
a, b = vals[0]
f = lambda x: exp_mod(x, a ,b)
fqs(f, xs, ys)
```

Out[7]: 2.83624841693

$$\log(y) = \log(a \cdot b^x) = \underbrace{\log(a)}_{q} + x \cdot \underbrace{\log(b)}_{m}$$

$$\Rightarrow \quad a = e^q \quad \Rightarrow \quad a = np.exp(q)$$

$$b = e^m \quad \Rightarrow \quad b = np.exp(m)$$

In [6]:
```
# Lösung zu Aufgabe 1.2
log_ys = np.log(ys)
m, q = sp.stats.linregress(xs, log_ys)[:2]
a = np.exp(q)
b = np.exp(m)
f = lambda x: a*b**x

# def f(x):
#     return a*b**x

fqs(f, xs, ys)
```

Out[6]: 1540.63083552