Váhy

Vážený průměr (váhy w_i nemusí být normalizované)

$$\overline{x} = \frac{\sum_i x_i w_i}{\sum_i w_i}$$

Známe x_i (nezávislé) s chybami σ_i. **Jaké máme volit váhy?**

Odvodíme pro 2 veličiny:

$$\overline{x} = wx_1 + (1-w)x_2$$

$$\sigma^2(\overline{x}) = \langle(\overline{x} - \langle x\rangle)^2\rangle = \langle(w\Delta x_1 + (1-w)\Delta x_2)^2\rangle = w^2\sigma_1^2 + (1-w)^2\sigma_2^2$$

Minimum nastane pro

$$w = \frac{1/\sigma_1^2}{1/\sigma_1^2 + 1/\sigma_2^2}, \quad 1 - w = w_2 = \frac{1/\sigma_2^2}{1/\sigma_1^2 + 1/\sigma_2^2}$$

Tedy nenormalizované váhy (a platí obecně, odvodí se pomocí Lagrangeových multiplikátorů):

$$w_i = \frac{1}{\sigma_i^2}$$

Ale problém může být, pokud neznáme σ_i přesně.

Máme-li aspoň trochu spolehlivou informaci pro výpočet vah (např. počet kroků simulace), pak ji použijeme, bude to lepší než $1/\sigma^2$.

Průměrování nezávislých měření

Máme data x_i s odhady standardních chyb σ_i.

1. Známé váhy. Např. (nenormalizováno) $w_i \propto n_i$, tedy každé x_i vzniklo zpracováním mnoha nezávislých měření, ale n_i neznáme nebo data jsou časově korelovaná, např. w_i = čas v simulaci.

$$\overline{x} = \frac{\sum_{i=1}^{m} w_i x_i}{\sum_{i=1}^{m} w_i}, \quad \sigma = \frac{\sqrt{\sum_{i=1}^{m} w_i^2 \sigma_i^2}}{\sum_{i=1}^{m} w_i}$$

Nepoužívat $w_i \approx 1/\sigma_i^2$, jsou-li w_i známy!

Příklad.

N_{cyc}	$Q(\sigma_Q)$
2000	201(14)
2000	191(17)
4000	196(11)

2. Neznámé váhy. Pak $w_i = 1/\sigma_i^2$ (jsou-li σ_i dost přesná) a z výše uvedeného vzorce:

$$\overline{x} = \frac{\sum_{i=1}^{m} x_i/\sigma_i^2}{\sum_{i=1}^{m} 1/\sigma_i^2}, \quad \sigma = \frac{1}{\sqrt{\sum_{i=1}^{m} 1/\sigma_i^2}}$$

Někdy pomůže σ_i resp. w_i vyrovnat.

Příklad.

$Q(\sigma_Q)$
201(14)
191(17)
197(11)

Průměrování nezávislých měření

Příklad. Dvě ekvivalentní MC simulace daly následující hodnoty parametru uspořádání: $\theta_1 = 2.17(33), \theta_2 = 2.32(31)$. Po změně zkušebního posunutí byla provedena třetí simulace, která dala $\theta_3 = 2.25(22)$. Vypočtěte nejlepší odhad parametru uspořádání.

mmpc6.mw a
mmpc6/6a-data-merging.py

Stejné váhy pro měření 1 a 2:

$$\theta_{12} = \frac{2.17 + 2.32}{2} = 2.245$$

$$\sigma_{12} = \frac{\sqrt{0.33^2 + 0.31^2}}{2} = 0.2264$$

Váhy odvozené z nejistot pro kombinaci výsledku 12 a měření 3:

$$\theta_{123} = \frac{2.245/0.2264^2 + 2.25/0.22^2}{1/0.2264^2 + 1/0.22^2} = 2.248$$

$$\sigma_{123} = \frac{1}{\sqrt{1/0.2264^2 + 1/0.22^2}} = 0.158$$

$$\underline{\theta = 2.25(16)}$$

Je-li málo dat. . .

3. Málo dat. Data vznikla z výběrů po n_i kouscích, pro které jsme spočítali průměry (označeny nyní x_i) a odhady standardních chyb průměrů (σ_i) pomocí obvyklých vzorců. Pak

$$\overline{x} = \frac{\sum_{i=1}^{m} n_i x_i}{\sum_{i=1}^{m} n_i}, \quad \sigma = \sqrt{\frac{\sum_{i=1}^{m} n_i(n_i-1)\sigma_i^2 + \sum_{i=1}^{m} n_i(x_i-\overline{x})^2}{\left(\sum_{i=1}^{m} n_i - 1\right)\sum_{i=1}^{m} n_i}}$$

jsou stejné, jako kdybychom všechna původní data spojili.
Pro velké n_i přejde v metodu **1.**

Příklad. Podle dat v příkladu s jezevčíky jsme spočítali:

$$x = [12.1, 20, 15.1, 20.8, 19.7] \;:\; \overline{x}_5 = 17.54 \pm 1.68$$
$$y = [18.9, 10.1, 12.1, 9.2, 12.4, 16.7, 12.7] \;:\; \overline{y}_7 = 13.16 \pm 1.31$$

Vypočtěte nejlepší odhad spodní výšky různými metodami.

1. $w_i = n_i$: 14.983 ± 1.040
2. $w_i = 1/\sigma^2$: 14.812 ± 1.036
3. 14.983 ± 1.185 (to samé vyjde po spojení dat = optimální postup)

Metoda nejmenších čtverců

$\vec{x}_i$ = nezávisle proměnné, typicky $\vec{x}_i \in \mathbb{R}^n$ (lze zobecnit)

y_i = závisle proměnné (reálná čísla)

y_i jsou nezávislé náhodné veličiny se známým σ_i

σ_i = standardní nejistota hodnoty y_i (neboli váhy $w_i = 1/\sigma_i^2$)

$f_{\vec{a}}(\vec{x})$ = funkce (zvaná „model“) schopná vystihnout data $(\vec{x}_i, y_i)$

$\langle y_i \rangle = f_{\vec{a}}(\vec{x}_i)$

$\vec{a}$ = parametry (p hodnot zapíšeme jako vektor), $p \leq n$, nejlépe $p \ll n$

Parametry $\vec{a}$ budeme hledat z podmínky minima součtu kvadrátů odchylek (S^2),kde

$$S^2 = \sum_{i=1}^{n} w_i \left[f_{\vec{a}}(\vec{x}_i) - y_i \right]^2 \equiv \sum_{i=1}^{n} \left[\frac{f_{\vec{a}}(\vec{x}_i) - y_i}{\sigma_i} \right]^2$$

hodnota $\min_{\vec{a}} S^2$ se nazývá reziduální součet čtverců (*residual sum of squares*)

Věta (Gauss–Markov): pro funkci $f_{\vec{a}}$ lineárně závisející na $\vec{a}$ je metoda nejmenších čtverců:

- **B**est (dává nejmenší varianci odhadnutých $\vec{a}$)
- **L**inear (= předpoklad)
- **U**nbiased ($\langle \vec{a} \rangle$ je správné)
- **E**stimate.

$n - p$ se nazývá „počet stupňů volnosti“ a často se značí ν (řecké „ný“)

Není-li úloha singulární, dále platí: $\langle S^2 \rangle = n - p$

Pro „dost dobře“ rozmístěné body x_i také $\lim_{n\to\infty} S^2 = n - p$ (posouzení kvality fitu)

Metoda nejmenších čtverců

Příklad. Pro $f_a(x) = a$ (konstanta) a $\sigma_i = 1$ najděte odhad parametru a

$$S^2 = \sum_{i=1}^{n} (a - y_i)^2 = na^2 - 2a \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} y_i^2$$

$$\frac{\mathrm{d}S^2}{\mathrm{d}a} = 2na - 2\sum_{i=1}^{n} y_i \stackrel{!}{=} 0 \quad \Rightarrow \quad a = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Výsledkem **fitování** (korelace, regrese, prokládání) je:

- odhad $\vec{a}$
- odhad standardních (směrodatných) chyb
- odhad korelací mezi parametry
- odhad hodnoty nějaké funkce $g(\vec{a})$ (vč. odhadu standardní chyby)

Příklady lineárního a nelineárního fitování viz Maple a Python →

mmpc6/mw b
mmpc6/6b-simple-fit.py

Metoda nejmenších čtverců – linearizace

Nechť $\vec{a}_0$ je přesná (hledaná) hodnota parametrů. Pro každé $\vec{x}$ rozvineme:

$$f_{\vec{a}}(\vec{x}) \approx f_{\vec{a}_0}(\vec{x}) + \sum_{j=1}^{p} \Delta a_j f_j(\vec{x}), \quad f_j(\vec{x}) = \frac{\partial f_{\vec{a}_0}(\vec{x})}{\partial a_j}$$

kde $\vec{a} = \vec{a}_0 + \Delta\vec{a}$.

Pokud jsou změny parametrů $\vec{a}$ malé, bez újmy na obecnosti stačí studovat lineární model

$$f_{\vec{a}}(\vec{x}) = \sum_{j=1}^{p} a_j f_j(\vec{x}),$$

kde $\{f_j(\vec{x})\}_{j=1}^{p}$ je (obecně neortogonální) báze.

Terminologická poznámka

Toto je „lineární regrese“, „lineární fit“, „lineární model“: používám lineární kombinaci bázových funkcí s neznámými koeficienty.

Tyto termíny mohou v užším smyslu znamenat fitování pomocí lineární funkce parametrů $f(\vec{x}) = a + \sum b_i x_i$ resp. v 1D jen $f(x) = a + bx$.

Lineární model

$$f_{\vec{a}}(\vec{x}) = \sum_{j=1}^{p} a_j f_j(\vec{x})$$

Nechť data y_i jsou nezávislé náhodné proměnné obecně s různými směrodatnými odchylkami σ_i. Data s chybami pak můžeme zapsat jako přesná hodnota + náhodná proměnná:

$$y_i = \sum_{j=1}^{p} a_j f_j(\vec{x}) + \Delta y_i, \quad \langle \Delta y_i \rangle = 0, \quad \langle \Delta y_i \Delta y_j \rangle = \sigma_i^2 \delta_{ij}$$

Následující **součet čtverců**:

$$S^2 = \sum_{i=1}^{n} \left[\frac{\sum_{j=1}^{p} a_j f_j(\vec{x}_i) - y_i}{\sigma_i} \right]^2 \qquad (1)$$

budeme minimalizovat, tedy spočítáme derivaci a položíme = 0:

$$\frac{\partial S^2}{\partial a_k} = 2 \sum_{i=1}^{n} \frac{f_k(\vec{x}_i)}{\sigma_i} \left[\frac{\sum_{j=1}^{p} a_j f_j(\vec{x}_i) - y_i}{\sigma_i} \right] = 0 \quad (2)$$

Definujeme: matice $p \times n$

$$F_{ki} = \frac{f_k(\vec{x}_i)}{\sigma_i}, \quad Y_i = \frac{y_i}{\sigma_i}, \quad \langle \Delta Y_i \Delta Y_j \rangle = \delta_{ij}$$

$$A_{kj} = \sum_{i=1}^{n} F_{ki} F_{ji}, \quad b_k = \sum_{i=1}^{n} F_{ki} Y_i$$

Rovnice (2) je pak:

$$\sum_{j=1}^{p} A_{kj} a_j - b_k = 0$$

Maticově ($\sum_1^p = \cdot$, $\sum_1^n = \blacksquare$):

$$A \cdot \vec{a} - \vec{b} = 0, \text{ kde } A = F \blacksquare F^{\mathsf{T}}, \ \vec{b} = F \blacksquare \vec{Y}$$

Metoda nejmenších čtverců – kovarianční matice

$$A \cdot \vec{a} - \vec{b} = 0, \quad \text{kde } A = F \cdot F^{\mathrm{T}}, \quad \vec{b} = F \cdot \vec{Y}$$

$$\vec{a} = A^{-1} \cdot \vec{b} = A^{-1} \cdot F \cdot \vec{Y} \quad \text{a tedy i} \quad \Delta\vec{a} = A^{-1} \cdot F \cdot \Delta\vec{Y}$$

Korelace mezi odhadnutými parametry jsou dány **kovarianční** nebo též **varianční-kovarianční** maticí C. Matice C je symetrická, na diagonále jsou variance:

$$C_{ij} = \mathrm{Cov}(a_i, a_j) = \langle \Delta a_i \Delta a_j \rangle$$

Vzorec pro výpočet C odvodíme v maticovém formalismu:

$$\begin{aligned} C = \langle \Delta\vec{a} \Delta\vec{a}^{\mathrm{T}} \rangle &= \langle A^{-1} \cdot F \cdot \Delta\vec{Y} [A^{-1} \cdot F \cdot \Delta\vec{Y}]^{\mathrm{T}} \rangle \\ &= \langle A^{-1} \cdot F \cdot \Delta\vec{Y} \Delta\vec{Y}^{\mathrm{T}} \cdot F^{\mathrm{T}} \cdot (A^{-1})^{\mathrm{T}} \rangle \\ &= A^{-1} \cdot F \cdot \langle \Delta\vec{Y} \Delta\vec{Y}^{\mathrm{T}} \rangle \cdot F^{\mathrm{T}} \cdot A^{-1} \\ &= A^{-1} \cdot F \cdot F^{\mathrm{T}} \cdot A^{-1} \\ &= A^{-1} \cdot A \cdot A^{-1} = A^{-1} \end{aligned}$$

Výsledkem fitování nejsou jen odhady variancí parametrů (na diagonále), ale i korelace (kovariance)!

- $\Delta\vec{a} \Delta\vec{a}^{\mathrm{T}}$ je vnější součin, též se značí $\Delta\vec{a} \otimes \Delta\vec{a}^{\mathrm{T}}$.
- $\langle \Delta\vec{Y} \Delta\vec{Y}^{\mathrm{T}} \rangle = \langle \Delta Y \otimes \Delta\vec{Y}^{\mathrm{T}} \rangle = \delta$ (jednotková matice).
- Pro matici M a vektor $\vec{v}$ platí: $(M \cdot \vec{v})^{\mathrm{T}} = \vec{v}^{\mathrm{T}} \cdot M^{\mathrm{T}}$.
- Matice A je symetrická: $A^{\mathrm{T}} = A, \ (A^{-1})^{\mathrm{T}} = A^{-1}$.

Metoda nejmenších čtverců

- Jsou-li všechna σ_i přesnými odhady standardních odchylek a máme dost bodů n, pak „reziduální směrodatná odchylka“ (*residual standard deviation*)

$$s = \sqrt{\frac{S^2}{n-p}}$$

  je blízko jedničky, „protože v rov. (1) je $n-p$ nezávislých čtverců, ⟨čtverec⟩ = 1“.

- Často σ_i neznáme, ale můžeme předpokládat, že všechna σ_i jsou stejná. Pak z rovnice $s = 1$ můžeme zpětně spočítat neznámé σ. Pokud S^2 bylo počítáno se $\sigma_i = 1$ (což dělá většina softwaru), platí

$$\sigma_i^{\text{odhad}} = s$$

  Kovarianční matice je pak počítána jako se $\sigma_i = \sigma_i^{\text{odhad}}$.

- Jsou-li funkce f_j kolmé*, pak A je diagonální a parametry nejsou korelované. Toto je často obtížné zajistit v praxi pro nelineární (ale lokálně linearizovatelné) odhady.

*vzhledem ke skalárnímu součinu $f_1 \cdot f_2 = \sum_{i=1}^{n} f_1(\vec{x}_i) f_2(\vec{x}_i)$ pro funkce f_1, f_2

Chyba funkce parametrů

Potřebujeme spočítat $g(\vec{a})$ a odhad jeho chyby

$$g_{\vec{a}} \approx g_{\vec{a}_0} + \sum_{j=1}^{p} \Delta a_j g_j(\vec{x}), \quad g_j = \frac{\partial g_{\vec{a}_0}}{\partial a_j} \tag{3}$$

- $\vec{g}$ je gradient funkce g podle parametrů
- lze interpretovat jako „citlivost“ na chybu v parametru

$$\sigma(g_{\vec{a}})^2 = \langle (g_{\vec{a}} - g_{\vec{a}_0})^2 \rangle = \left\langle \sum_{ij} \Delta a_i g_i \Delta a_j g_j \right\rangle = \sum_{ij} g_i \mathrm{Cov}(a_i, a_j) g_j = \sum_{ij} g_i A^{-1}_{ij} g_j$$

- Příklady $g(\vec{a})$:
  - a_i (jeden z parametrů)
  - $\int_{x_0}^{x_1} f(x)\mathrm{d}x$ (např. entalpie z tepelné kapacity)
  - $f(x_0)$ pro zvolené x_0 (vč. extrapolace)
  - kořen rovnice $f(x) = y_0$

Příklad 1: chyby dat jsou neznámé a všechny stejné

Analyzujeme rozkladnou reakci 1. řádu. Koncentrace jako funkce času jsou:

t [s]	0	10	20	30	40	50	60	70	80	90	100	110	120
c [mM]	3.371	3.291	2.211	1.903	1.846	1.335	1.308	1.019	0.716	0.569	0.656	0.572	0.455

Vypočtěte koncentraci v čase $t = 300\,\mathrm{s}$ včetně odhadu nejistoty. Předpokládejte, že **relativní** nejistoty koncentrací jsou stejné.

$c(300\,\mathrm{s}) = 19(4)\,\mu\mathrm{M}$

- To znamená, že **absolutní** chyby $\ln c$ jsou stejné, budeme tedy fitovat $\ln c$ lineární funkcí

$$\ln c(t) = \ln c_0 - kt$$

- Dále viz software:

  mmpc6.mw: c 3× linear fit with error analysis, Sec. 1
  mmpc6/6c-1-linfit.py

- Kdybychom výsledek porovnávali s přesným výsledkem pomocí Studentova t-testu, použijeme $\nu = n - p = 13 - 2 = 11$.

Fitování Maple 1: chyby dat jsou neznámé a všechny stejné

Neznáme chyby (standardní nejistoty) dat σ_i, ale předpokládáme, že jsou pro všechna data stejné. Do funkce Fit nezadám volitelný vektor weights (tj. všechny váhy jsou jedničky).

Syntax: Fit($f_{\vec{a}}(x)$, $\vec{x}$, $\vec{y}$, x, output=[key,key,...]), kde key jsou:

neúplné, viz Maple manual pro další možnosti

parametervalues = vypočtené parametry a_j, $j = 1..p$

parametervector = to samé ve formě vektoru

leastsquaresfunction = $f_{\vec{a}}(x)$ s dosazenými vypočtenými parametry (x je formální reálný parametr)

residualsumofsquares = S^2 podle rov. (1) s $\sigma_i = 1$

residualstandarddeviation = $s = \sqrt{S^2/(n-p)}$
= odhad směrodatných odchylek σ_i dat (všechna σ_i jsou stejná)

standarderrors = odhady standardních chyb parametrů, $\sigma(a_j)$, $j = 1..p$

residuals = rozdíly nezávisle proměnných od nafitovaných hodnot, $y_i - f_{\vec{a}}(x_i)$, $i = 1..n$

variancecovariancematrix = kovarianční matice, $\mathrm{Cov}(a_i, a_j)$, $i, j = 1..p$ (jen pro lineární modely)

Fitování Python 1: chyby dat jsou neznámé a všechny stejné

Neznáme chyby (standardní nejistoty) dat σ_i, ale předpokládáme, že jsou pro všechna data stejné.

```
from scipy.optimize import curve_fit # jsou i jiné možnosti
X = np.array([X-DATA])
Y = np.array([Y-DATA])
def model(x, a, b): # a,b jsou parametry, model nemusí být lineární
    return FUNKCE(x,a,b)

parms, cov = curve_fit(model, X, Y)
a,b = parms # vypočtené parametry
# cov je varianční-kovarianční matice
sigma_a = np.sqrt(cov[0][0]) # odhad standardní chyby parametru a
sigma_b = np.sqrt(cov[1][1]) # odhad standardní chyby parametru b
g = np.array([CITLIVOSTI])
sigma = np.sqrt(g @ cov @ g) # odhad standardní chyby funkce g(a,b)
```

Reziduální směrodatná odchylka není v `curve_fit` dostupná, ale lze ji spočítat (viz dále).

Příklad 2: chyby dat neznáme, ale váhy nejsou stejné

Měřili jsme tepelnou kapacitu vzorku jako funkci teploty t. Naměřili jsme 8 bodů (t, C_p) na přístroji DSC-1 (pro $t < 55\,°C$) and 2 body na přesnějším přístroji DSC-2 (pro $t = 55\,°C$ a $60\,°C$). Podle literatury je DSC-2 asi dvakrát přesnější (tj. očekávaná nejistota měření je poloviční ve srovnání s DSC-1). Předpokládejte že nejistoty měření jsou náhodné a nekorelované.

1) Nafituje data na lineární funkci a zobrazte data a směrodatnou odchylku dat.

2) Vypočtěte nejpřesnější hodnotu za teploty $t = 25\,°C$.

t [°C]	5	10	15	20	25	30	35	40	45	50	55	60
C_p [J/K]	3.615	3.408	4.107	4.404	5.937	5.565	7.023	6.577	7.632	8.31	9.133	9.833
w [rel.]	1	1	1	1	1	1	1	1	1	1	4	4

Poloviční nejistota měření znamená čtyřnásobnou váhu.

$C_p(25\,°C) = 5.43(14)$ J/K

- Dále viz software:

  mmpc6.mw: c 3× linear fit with error analysis, Sec. 2
  mmpc6/6c-2-linfit.py

- Kdybychom výsledek porovnávali s přesnou referencí pomocí Studentova t-testu, použijeme $\nu = n - p = 12 - 2 = 10$.

Fitování Maple 2: chyby dat neznáme, ale váhy nejsou stejné

Víme, že některá data jsou kvalitnější, což vyjadřujeme váhami w_i, $i = 1..n$, ale neznáme nejistoty σ_i; víme jen, že $\sigma_i/\sigma_j = \sqrt{w_j/w_i}$.

Syntax: Fit($f_{\vec{a}}(x)$, $\vec{x}$, $\vec{y}$, x,weights = $\vec{w}$, output=[key,key,...])

Modifikovaný vzorec s váhami pro S^2 je:

$$\text{residualsumofsquares} = S^2 = \sum_{i=1}^{n} w_i \left[\sum_{j=1}^{p} a_j f_j(\vec{x}_i) - y_i \right]^2$$

$$\text{residualstandarddeviation} = s = \sqrt{S^2/(n-p)}$$

⇒ odhadnuté směrodatné odchylky (standardní chyby) původních dat y_i jsou (pokud nás zajímají):

$$\sigma_i = \frac{s}{\sqrt{w_i}} = \frac{\text{residualstandarddeviation}}{\sqrt{\text{weight[i]}}}$$

Fitování Python 2: chyby dat neznáme, ale váhy nejsou stejné

Víme, že některá data jsou kvalitnější, což vyjadřujeme relativními směodatnými odchylkami σ_i, $i = 1..n$. Absolutní hodnoty neznáme, víme jen, že váhy w_i jsou úměrné $1/\sigma_i^2$.

```
...
sig = np.array([REL STD ODCHYLKY DAT])
parms, cov = curve_fit(model, X, Y, sigma=sig, absolute_sigma=False)
...
SS = np.sum( ((model(X,a,b)-Y)/sig)**2 ) # přeškálovaný reziduální součet čtverců
p = POČET PARAMETRŮ
s = np.sqrt(SS/(len(X)-p))
s*sig # vypočtená reziduální směrodatná odchylka dat (array)
```

Chyba funkce parametrů se počítá stejně jako v případu 1.

Fitování 3: nejistoty dat jsou známé a spolehlivé

Metodou MC simulace byly vypočteny statistické váhy $W = e^S$, kde S je reziduální entropie na vrchol mřížky, pro periodické vzorky ledu I_h v závislosti na počtu molekul:

N	360	2880	9720	23040	45000	77760	184320
W	1.51898342	1.50906444	1.50796203	1.50767991	1.50757977	1.50753663	1.50749468
σ	0.00001264	0.00000887	0.00000650	0.00000566	0.00000466	0.00000405	0.00000348

Chyby jsou výsledkem dlouhých MC simulací a jsou tedy spolehlivé. Teorie dává extrapolační vztah:

$$W(N) = a + \frac{b}{N} + c\frac{\ln(N)}{N}$$

Stanovte termodynamickou limitu $a = W(\infty)$.

$W = 1.507465151(256)$

Pozn.: Grafy jsou přehlednější, použijeme-li jako nezávislou proměnnou $x = 1/N$:

$$W(x) = W(0) + bx - cx\ln x$$

- Dále viz software:

  mmpc6.mw: c 3× linear fit with error analysis, Sec. 3
  mmpc6/6c-3-linfit.py

- Kdybychom výsledek porovnávali s přesnou hodnotou, použijeme Gaussovo normální rozdělení (počet stupňů volnosti je velký).

Fitování Maple 3: nejistoty dat jsou známé a spolehlivé

Máme data y_i s přesnými odhady směrodatných chyb σ_i, ale nemáme příliš mnoho dat.
Tj. chyby odvozené z fitování těchto dat bez znalosti σ_i resp. jen se znalostmi vah neboli poměrů σ_i/σ_j by nebyly spolehlivé.

Vytvoříme si vektor $\vec{w}$, kde $w_i = 1/\sigma_i^2$, a postupujeme jako na předchozím slidu. Platí:

residualstandarddeviation $= s = \sqrt{S^2/(n-p)}$ **má být blízko 1**
tím blíže, čím více dat máme, zhruba $1 \pm (n-p)^{-1/2}$

- Pokud $s \gg 1$, máme nevhodný model (např. polynom nedostatečného stupně), případně podhodnocené odhady σ_i.
- Pokud $s < 1$, můžeme mít zbytečně složitý model (fitujeme šum), případně nadhodnocené odhady σ_i.
- Dle předpokladu jsou σ_i přesná, proto přeškálujeme jak odhady standardních chyb parametrů:

$$\sigma(a_j) = \frac{\text{standarderrors[j]}}{\text{residualstandarddeviation}}$$

tak i kovarianční matici

$$\mathrm{Cov}(a_i, a_j) = \frac{\text{variancecovariancematrix[i][j]}}{\text{residualstandarddeviation}^2}$$

Fitování Python 3: nejistoty dat jsou známé a spolehlivé

Máme data y_i s přesnými odhady směrodatných chyb σ_i, ale nemáme příliš mnoho dat.

Tj. chyby odvozené z fitování těchto dat bez znalosti σ_i resp. jen se znalostmi vah neboli poměrů σ_i/σ_j by nebyly spolehlivé.

mmpc6/6c-3-linfit.py

```
...
sig = np.array([SMĚRODATNÉ ODCHYLKY DAT])
parms, cov = curve_fit(model, X, Y, sigma=sig, absolute_sigma=True)
...
SS = np.sum( ((model(X,a,b)-Y)/sig)**2 ) # reziduální součet čtverců
p = POČET PARAMETRŮ
s = np.sqrt(ss/(len(X)-p)) # pro dost dat by mělo být okolo 1
s*sig # vypočtená reziduální směrodatná odchylka dat (array)
```

Hodnocení případů $s \gg 1$ a $s < 1$ viz předchozí stránka.

Pokud dle předpokladu jsou σ_i přesná, odhady standardních chyb parametrů již neškálujeme jako u Maplu. Kdybychom nevěřili přesnosti σ_i a chtěli odhady nejistot založit na rozptylu dat, znásobili bychom $\sigma(a_j) \times s$ a $\mathrm{Cov}(a_i, a_j) \times s^2$; výsledek by byl stejný jako v případu 2 a parametru `absolute_sigma=False`.

Metoda nejmenších čtverců – chyby MC metodou

- Minimalizujeme $S^2 \Rightarrow$ získáme $\vec{a}_0$ a $g(\vec{a}_0)$.
- Pro $k = 1..m$ provedeme:
  - Vyrobíme falešná data

$$y_i^{(k)} = f_{\vec{a}_0}(\vec{x}_i) + \sigma_i u,$$

    kde u je náhodné číslo s normalizovaným normálním rozdělením ($\sigma = 1, \mu = 0$).

    Chyby σ_i dat y_i známe; pokud ne, použijeme $\sigma_i = \sqrt{S^2/(n-p)}$.
  - Vypočteme metodou nejmenších čtverců parametry $\vec{a}^{(k)}$.
  - Vypočteme $g(\vec{a}^{(k)})$.
- Výsledky $\vec{a}^{(k)}$ a $g(\vec{a}^{(k)})$ pro $k = 1..m$ zpracujeme jako nezávislá data a dostaneme odhad standardní chyby $\sigma(g)$.

Předchozí příklad touto metodou pro $m = 999$ vzorků falešných dat (zeleně = „přesně“)

$$\begin{aligned} a &= 1.50746515(257) \quad (256) \\ b &= 2.8786(720) \quad\quad (724) \\ c &= 0.2154(122) \quad\quad (122) \end{aligned}$$

Metoda nejmenších čtverců – co je uvnitř softwaru

Lineární model: řešitelný metodou lineární algebry, nebývají problémy (pokud jsou, stačí ortonormalizovat bázi)

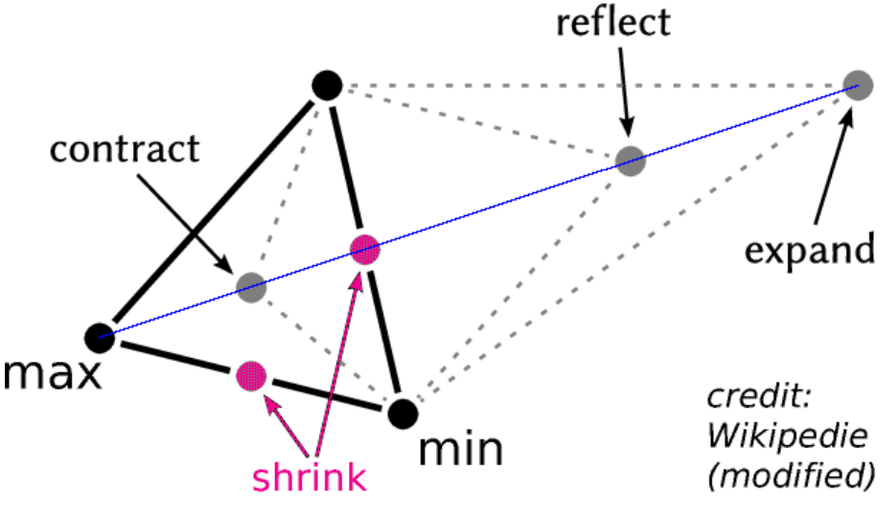
Nelineární model:

Problém 1: můžeme mít více lokálních minim,
některá pro nevlastní hodnoty parametrů (= divergence)

Problém 2: dlouhá zakřivená údolí při minimalizaci (pomalé)

Minimalizace nelineární funkce mnoha proměnných:

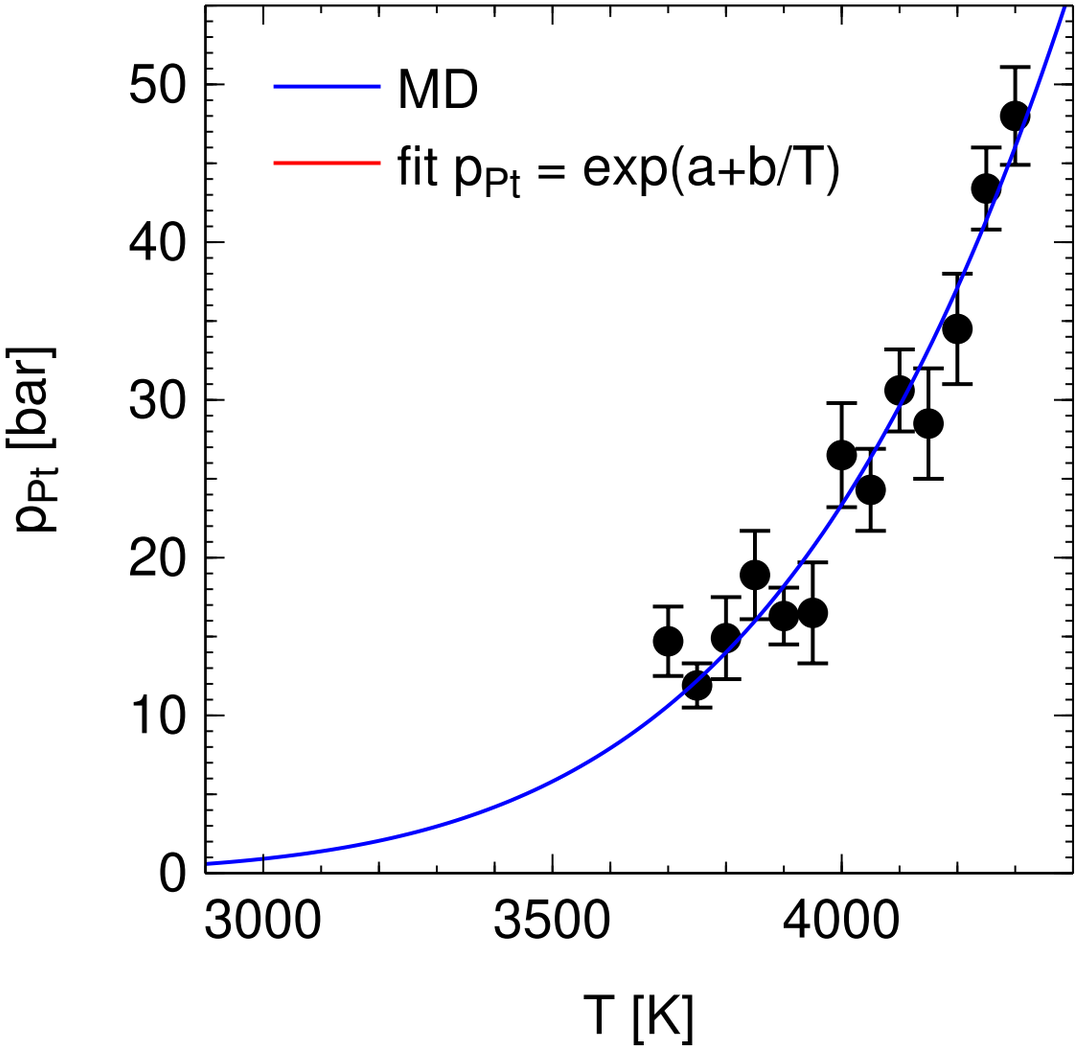
- hledání na mřížce (na začátku)
- Monte Carlo hledání (na začátku)
- metoda největšího spádu (steepest descent, greedy)
- konjugované gradienty
- améba (Nelder–Mead; *amoeba, downhill simplex*) →
- (Gaussova–)Newtonova metoda (jsme-li blízko řešení)
- (Levenbergova–)Marquardtova metoda (kombinace Newton, gradient, tlumení)
- simulované žíhání



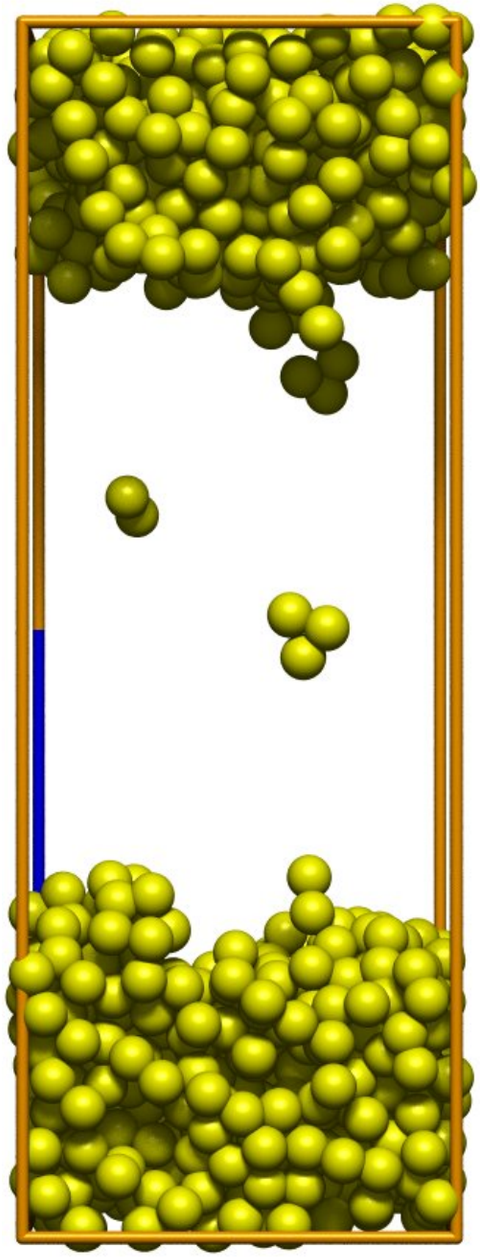
credit: Wikipedie (modified)

Příklad z praxe

Při simulaci modelu platiny ve slab-geometrii byla získána data pro tlak ve směru kolmém na vrstvu:



T/K	p/bar	σ/bar
3700	14.7	2.2
3750	11.9	1.4
3800	14.9	2.6
3850	18.9	2.8
3900	16.3	1.8
3950	16.5	3.2
4000	26.5	3.3
4050	24.3	2.6
4100	30.6	2.6
4150	28.5	3.5
4200	34.5	3.5
4250	43.4	2.6
4300	48.0	3.1



Vypočtěte bod varu platiny za tlaku 1 bar a odhadněte chybu.

Řešení

Budeme předpokládat platnost Clausiovy–Clapeyronovy rovnice a konstantní výparnou entalpii.

$p = \exp(a + b/T)$ (vhodnější vzhledem k chování Δp) $\quad \ln p = a + b/T$ (linearizované)

kde a a b jsou konstanty, které budeme fitovat. Pak stanovíme hodnotu funkce g, která je řešením rovnice $\ln p = a + b/T$ pro $p = 1$ bar.

(σ): věříme uvedeným σ
(σ): věříme vahám, ale σ počítáme z reziduálního součtu čtverců

- Přímé fitování na $p = \exp(a + b/T)$:
  $s = 1.067$, $T_{\text{vap}} = 3021(55)(59)$ K
- Fitujeme $\ln(p)$ vs. $1/T$ (lineární regrese):
  $s = 1.081$, $T_{\text{vap}} = 2992(53)(57)$ K stejné jako lineární model $\ln(p)$ vs. T
- „Neznáme“ standardní chyby, jen předpokládáme, že $\sigma_i(p)$ jsou stejné ($p = \exp(a + b/T)$):
  $T_{\text{vap}} = 3015(74)$ K předpoklad, že $\sigma_i(\ln p)$ jsou stejné, je nevhodný
- Data získána ze stejně dlouhých trajektorií $\Rightarrow$ chyby lze vyrovnat ($p = \exp(a + b/T)$):
  $s = 1.138$, $T_{\text{vap}} = 2965(63)(72)$ K = nejlepší postup
- Viz výše, fitujeme $\ln(p)$ vs. T:
  $s = 1.118$, $T_{\text{vap}} = 2933(60)(67)$ K

Závěr: $T_{\text{vap}} = 2965(70)$ K, kde chyba 70 je 63 zaokrouhleno nahoru s přihlédnutím k 72

Další příklad

Nafitujte data na vhodnou funkci $f(x)$ a vypočtěte řešení rovnice $f(x_0) = 1$ vč. odhadu chyby.

12.49(5), věříme-li chybám σ_i a modelu;
12.49(7), nejsou-li chyby tak spolehlivé

x	y	σ
2	4.001	0.014
3	3.424	0.013
4	3.039	0.011
5	2.710	0.010
6	2.482	0.009
7	2.208	0.008
8	1.985	0.008
9	1.749	0.007
10	1.528	0.007

Pokud neumíme řešit rovnici $f(x) = y$, lze „citlivosti“ g_i (viz rov. (3)) na parametrech získat ze vzorce pro derivaci implicitní funkce. Píšu g (je to funkce parametrů) místo x, abych se přiblížil značení z rov. (3):

$$f(a_i + \mathrm{d}a_i, g + \mathrm{d}g) = y$$

$$\frac{\partial f}{\partial a_i}\mathrm{d}a_i + \frac{\partial f}{\partial g}\mathrm{d}g = 0$$

$$g_i \equiv \frac{\partial g}{\partial a_i} = -\frac{\partial f}{\partial a_i} / \frac{\partial f}{\partial g}$$

mmpc6.mw f Complex fitting example
mmpc6/6f-complex-fitting-example.py

Fitování v Excelu a LibreOfficu I

Excel a LibreOffice mají obecnou funkci pro lineární regresi LINEST v české lokalizaci LINREGRESE

Funkce LINEST fituje data $\vec{y}$ (n-vektor) na lineární funkci p vektorů $\vec{x}_j$, $j = 1..p$:

$$\vec{y} = a_0 + \sum_{j=1}^{p} a_j \vec{x}_j \tag{4}$$

Absolutní člen a_0 je volitelný, viz 3. argument funkce LINEST.

Funkce LINEST vrací hodnoty parametrů a_j vč. odhadů standardních chyb a $S/\sqrt{n-p}$ pro jednoduchou lineární regresi bez vah (váhy = 1).†

Příklad. Pro fitování na $a_1 + a_2 x + a_3 \ln x$, bázové vektory $\vec{x}_j$ jsou:

$$\begin{aligned} \vec{x}_1 &= (\vec{x})^0 = (1, 1, .., 1)^{\mathrm{T}} \quad \text{nebo verze s členem } a_0+ \\ \vec{x}_2 &= \vec{x} = (x_1, x_2, .., x_n)^{\mathrm{T}} \\ \vec{x}_3 &= \ln(\vec{x}) = (\ln x_1, \ln x_2, .., \ln x_n)^{\mathrm{T}} \end{aligned}$$

kde $\vec{x}$ je původní vektor nezávislých x-ů a funkce se interpretují člen po členu.

†AFAIK, kovarianční matice není dostupná. Po určité námaze ji lze vypočítat pomocí vzorců na str. 7–8 a maticových funkcí MMULT, MINVERSE.

Fitování v Excelu a LibreOfficu II

- Připrav sloupcové vektory $\vec{y}$ a $\vec{x}$

Neobsahuje všechny možnosti

- Připrav bázové vektory $\vec{x}_j$, $j = 1..p$
- Označ obdélníkové pole velikosti $(p+1) \times 4$ buněk pro výsledky
- Do první buňky tohoto pole napiš (naklikej, kde to jde)

  =LINEST(Y1:Yn, X1:X^pn, 0, 1)

V české lokalizaci nutno použít ; místo ,

  kde argumenty jsou:

  1. $\vec{y} = Y1 : Yn$ (sloupec)
  2. $\vec{x}_1..\vec{x}_p = X1 : X^p n$ je matice $p \times n$ (p sloupců)
  3. 0 znamená, že a_0+ se neuvažuje
  4. 1 znamená bohatý výstup

- Proveď „opičí hmat“ **Ctrl-Shift-Enter**

Výsledné odhady jsou ve formě pole $(p+1) \times 4$:

$\langle a_p \rangle$	$\langle a_{p-1} \rangle$	...	$\langle a_2 \rangle$	$\langle a_1 \rangle$	?
$\sigma(a_p)$	$\sigma(a_{p-1})$	...	$\sigma(a_2)$	$\sigma(a_1)$	n.a.
r^2	$S/\sqrt{n-p}$	n.a.	n.a.	n.a.	n.a.
F-value	$n-p$	n.a.	n.a.	n.a.	n.a.
?	$S^2 = \sum_i [f(x_i) - y_i]^2$	n.a.	n.a.	n.a.	n.a.

Pozor na opačné pořadí vypočtených parametrů!
r = korelační koeficient

Fitování v Excelu a LibreOfficu III

Jsou-li pro data $\vec{y}$ známy odhady standardních chyb $\vec{\sigma}$, nejprve připravíme sloupce obsahující:

$$\vec{y}' = \frac{\vec{y}}{\vec{\sigma}},\ \vec{x}'_j = \frac{\vec{x}_j}{\vec{\sigma}},\ j = 1..p$$

kde vektory se dělí prvek po prvku.

Dále je to stejné jako na předchozí stránce.

Opakuji, že hodnota $s = S/\sqrt{n-p}$ má být zhruba 1. Jsou-li známé hodnoty $\vec{\sigma}$ spolehlivější (založené na mnoha měřeních) než tato analýza ze $\nu = n-p$ stupňů volnosti, vydělte odhady chyb parametrů $\sigma(a_j)$ hodnotou s.

Příklad. Nafitujte následující data na funkci $a + bx + cx^2$:

x	−2.0	−1.8	−1.6	−1.4	−1.2	−1.0	−0.8	−0.6	−0.4	−0.2	0.0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
y	11.876	10.918	9.746	8.761	7.791	7.003	6.408	5.452	5.010	4.325	3.622	3.466	4.087	3.257	3.517	3.546	2.575	2.525	3.807	3.162	4.141
σ	0.70	0.66	0.62	0.58	0.54	0.50	0.46	0.42	0.38	0.34	0.30	0.34	0.38	0.42	0.46	0.50	0.54	0.58	0.62	0.66	0.70

$a = 4.02(13),\ b = -1.98(11),\ c = 0.99(9)$

mmpc6.mw g Linear fit example, to be compared to student+fit.ods / student+fit.xlsx
mmpc6/6g-student+fit.py