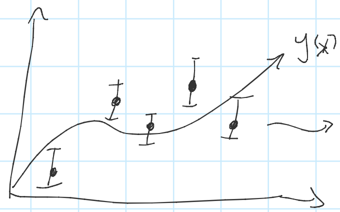
Week 4 - Monday

Monday, February 15, 2021 11:55 AM

Least-Squares fitting

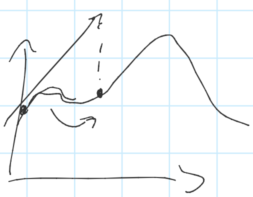


$\longrightarrow (x_i, y_i), \sigma_i$

$$P(y_i | y(x)) \propto e^{-\frac{1}{2}\left(\frac{y_i - y(x_i)}{\sigma_i}\right)^2}$$

$$P(\{y_i\} | y(\{x_i\})) \propto \prod_i e^{-\frac{1}{2}\left(\frac{y_i - y(x_i)}{\sigma_i}\right)^2}$$

for parameters in a model $y(x, \vec{c})$,
want to maximize P



try to maximize using:
- Gradient descent (follow gradients)
- Gauss-Newton (fit a paraboloid, travel to max.)
- Levenberg-Marquardt (combination of ↑) (more robust, not converge as fast as Gauss-N)
- Monte-Carlo Methods

Or, minimize: $\sum_i \left(\frac{y_i - y(x_i)}{\sigma_i}\right)^2 \equiv \chi^2$

Consider Linear case, $y(x) = \underline{a} + \underline{b}x$ min. w.r.t.

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

$$0 = \frac{\partial \chi^2}{\partial a} = -2\sum_i \frac{y_i - a - bx_i}{\sigma_i^2}$$

$$0 = \frac{\partial \chi^2}{\partial b} = -2\sum_i x_i \frac{y_i - a - bx_i}{\sigma_i^2}$$

more succinct? Write in terms of variance-weighted sums

$$S = \sum_i \frac{1}{\sigma_i^2}, \quad S_x = \sum_i \frac{x_i}{\sigma_i^2}$$

$$S_y = \sum_i \frac{y_i}{\sigma_i^2}$$

$$S_{xx} = \sum_i \frac{x_i^2}{\sigma_i^2} \quad S_{xy} = \sum_i \frac{x_i y_i}{\sigma_i^2}$$

$$\Delta = S S_{xx} - S_x^2$$

$$aS + bS_x = S_y$$

$$aS_x + bS_{xx} = S_{xy}$$

Sols

$$\boxed{\begin{aligned} a &= \frac{S_{xx} S_y - S_x S_{xy}}{\Delta} \\ b &= \frac{S S_{xy} - S_x S_y}{\Delta} \end{aligned}}$$

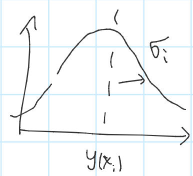
Unc. prop. of errors

$$\sigma_a^2 = S_{xx}/\Delta$$

$$\sigma_b^2 = S/\Delta$$

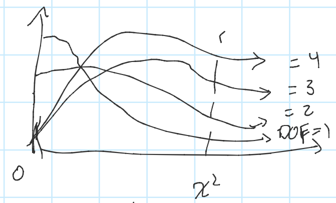
Goodness of fits:

"Know" $P(y_i)$



$P(\chi^2, \#DOF)$

$$\chi^2 = \sum_i \left(\frac{y_i - y}{\sigma_i}\right)^2$$



Was χ^2 drawn from ↑

$$Q = \text{Prob. of finding a } \underline{\text{worse}}\ \chi^2 = \underline{1 - \int_0^{\chi^2} d\chi'^2\, P(\chi'^2, DOF)}$$

scipy function for this

Q too close to 0 ⇒ bad fit

too close to 1 ⇒ Unreasonably good.

Connect to linear Eqs, SVD (applicable to linear least squares)

$$f(x) = c_1 f_1(x) + c_2 f_2(x) + \dots$$

$$A_{ij} = \frac{f_j(x_i)}{\sigma_i}, \quad A = \begin{pmatrix} \frac{f_1(x_1)}{\sigma_1} & \frac{f_2(x_1)}{\sigma_1} & \cdots \\ \frac{f_1(x_2)}{\sigma_2} & \ddots & \\ \vdots & & \end{pmatrix}$$

Vector with elements labeled by i

$$\chi^2 = \left|\sum_j A_{ij} c_j - y_i\right|^2 = \left|A\vec{c} - \vec{y}\right|^2$$

⇓

SVD solution is least-squares best fit! (minimizes χ^2)

$$\vec{c} = A^+ \vec{y} = V \Sigma^+ U^T \vec{y}$$

set $\frac{1}{0}$ elements to zero

Schematic proof:

Consider $\vec{c}'$,

$$\sqrt{\chi^2} = \left| A(\vec{c} + \vec{c}') - \vec{y} \right| = \left| (U \Sigma V^T)(V \Sigma^+ U^T) \vec{y} - \vec{y} + A\vec{c}' \right|$$

$$= \left| \underbrace{(\Sigma \Sigma^+ - 1) U^T \vec{y}}_{\text{purely in the null space}} - \underbrace{U^T A \vec{c}'}_{\text{purely in row space of } A} \right|$$

$\perp$

$\Rightarrow$ need $\vec{c}' = 0$ to maximize norm.

Ordinary Matrix language:

$$\chi^2 = |A\vec{c} - \vec{y}|^2 = (A\vec{c} - \vec{y})(A\vec{c} - \vec{y})^T$$

$$= \vec{c}^T A^T A \vec{c} - \vec{c}^T A^T \vec{y} - \vec{y}^T A \vec{c} + \vec{y}^T \vec{y}$$

$$0 = \frac{\partial \chi^2}{\partial \vec{c}^T} \quad \Bigg| \quad \frac{\partial \vec{c}}{\partial \vec{c}} = I, \quad \frac{\partial \vec{c}}{\partial \vec{c}^T} \Rightarrow \text{structure that transposes a vector}$$

$$\frac{\partial \chi^2}{\partial \vec{c}^T} = A^T A \vec{c} + (\vec{c}^T A^T A)^T - A^T \vec{y} - (\vec{y}^T A)^T$$

$$= 2A^T A \vec{c} - 2A^T \vec{y} = 0$$

$$(A^T A)\vec{c} = A^T \vec{y} \Rightarrow \boxed{\vec{c} = (A^T A)^{-1} A^T \vec{y}}$$

$$A = U \Sigma V^T$$

$$A^T = V \Sigma U^T$$

$$A^T A = V \Sigma^2 V^T$$

$$(A^T A)^{-1} = V \Sigma^{-2} V^T$$

$\frac{1}{0} \to 0$

$$\vec{c} = V \Sigma^{-2} V^T V \Sigma U^T \vec{y}$$

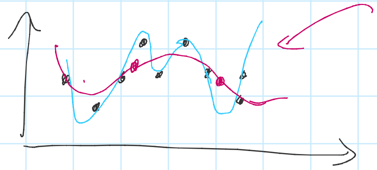
$$= V \Sigma^{-1} U^T \vec{y}$$

$$\boxed{\vec{c} = A^+ \vec{y}}$$

★ InterpolatedUnivariateSpline => "perfect" fit

Fit a spline! => ★ UnivariateSpline => "Smoothed" fit

Fit a spline! => Univariate Spline => "Smoothed" fit



Knots = # piecewise intersections