

1. Data Fitting

Things everyone seems know but me

Dong-Hee Kim

Gwangju Institute of Science and Technology

References

**“Everything you wanted to know about Data Analysis and Fitting
but were afraid to ask”**

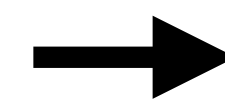
by A. Peter Young [[arXiv:1210.3781](https://arxiv.org/abs/1210.3781)].

Goals of data fitting

DATA

“N” DATA POINTS = $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$

UNCERTAINTY (ERROR BAR) = $\{\sigma_1, \sigma_2, \dots, \sigma_N\}$

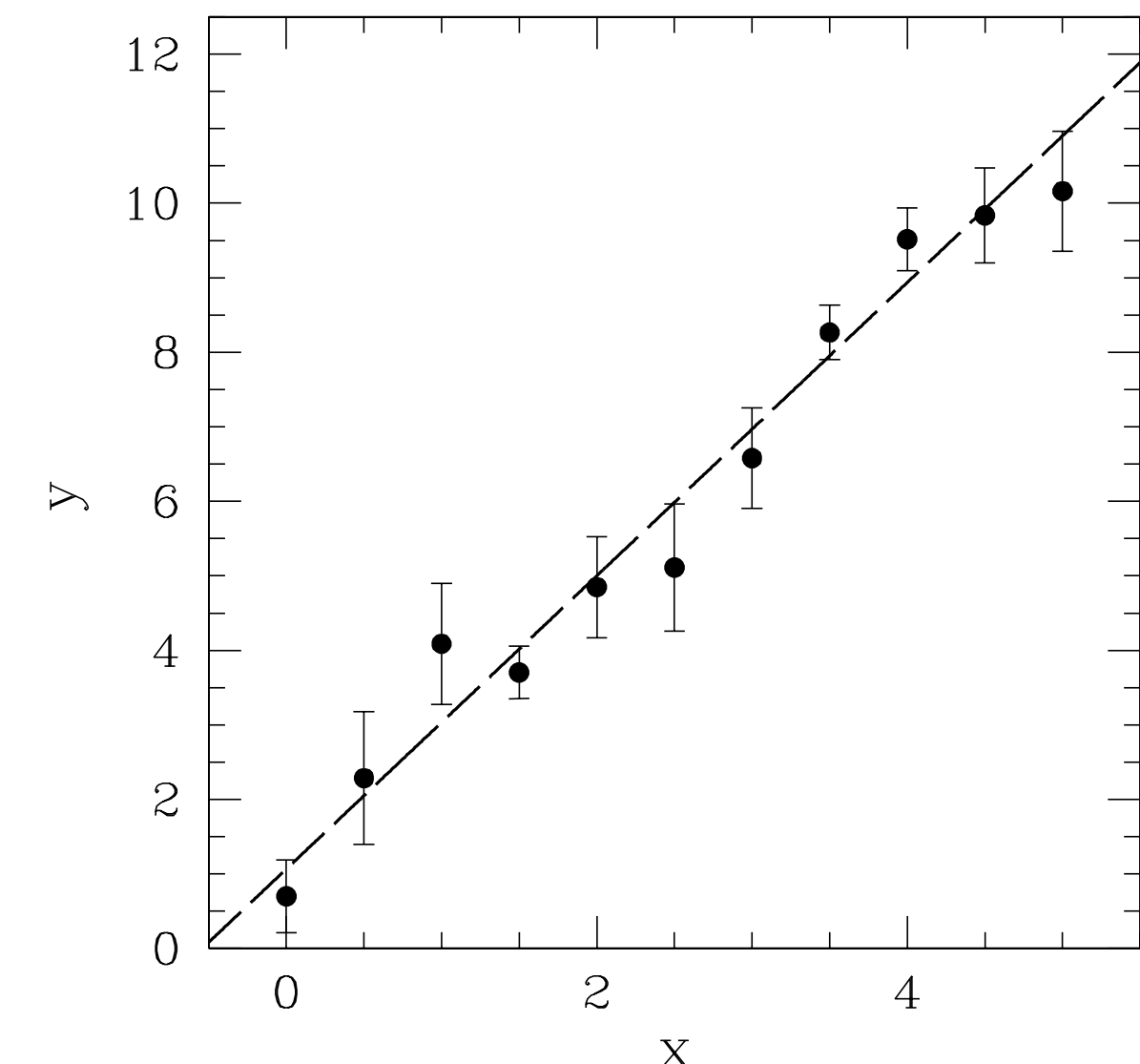


MODEL

$f(x)$

“M” FITTING PARAMETERS = $(a_0, a_1, \dots, a_{M-1})$

1. Find the fit parameters.
2. Provide error estimate on the fit parameters.
3. Provide a measure of how good the fit is.



Least-square fitting

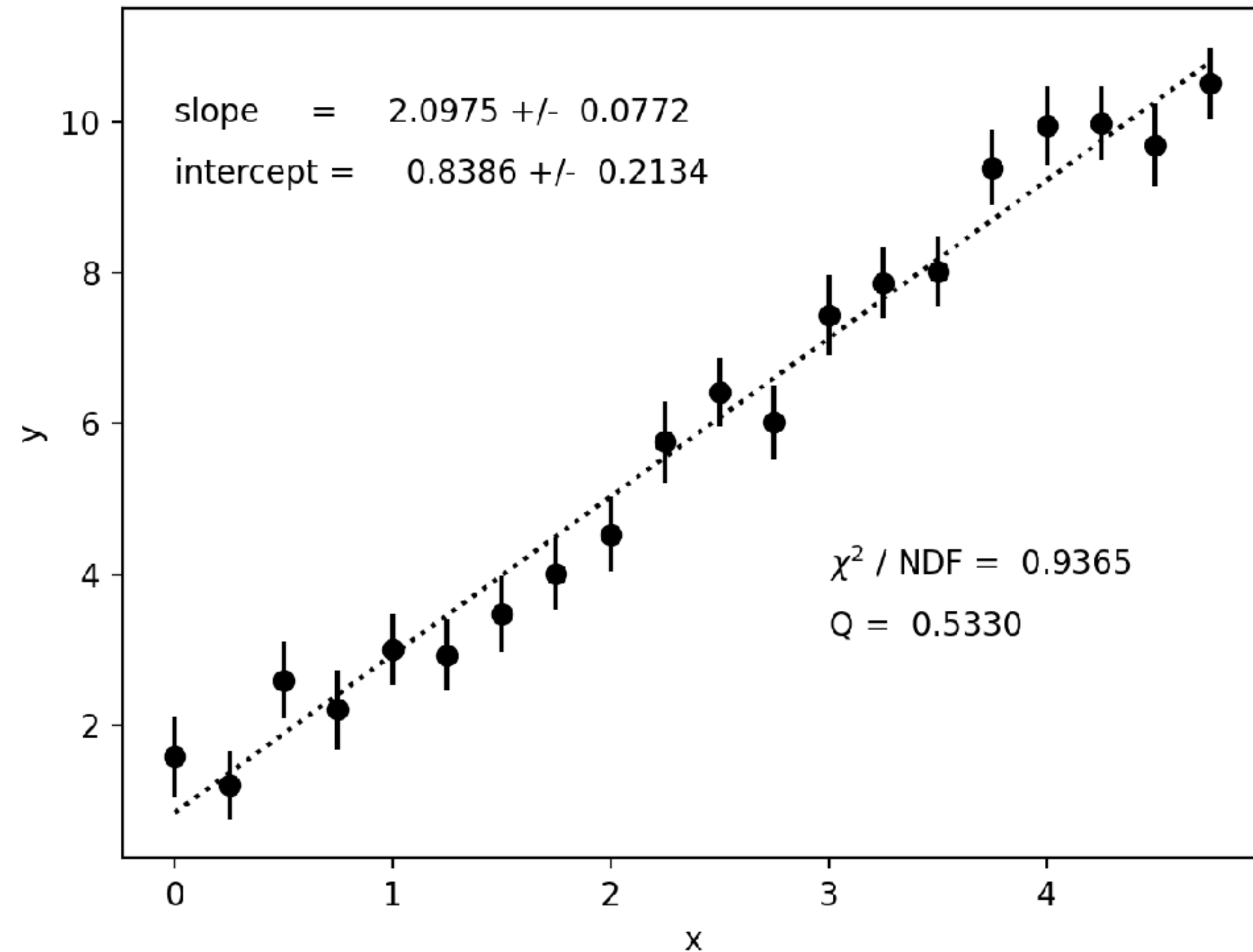
- The definition of the “best” fit is not unique.
- Probably the most popular one is:

$$\chi^2 = \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2 \longrightarrow \text{MINIMIZE!}$$

x	y	sigma
0.0	1.5891700001638700	0.5320185951347170
0.25	1.2046979031630900	0.45427229254919400
0.5	2.598484110379440	0.5077027054798450
0.75	2.2043375760987900	0.5205826529502230
1.0	3.002229375836470	0.4698090780151540
1.25	2.928655222350250	0.4773624121076510
1.5	3.4769844321649000	0.5018787752525470
1.75	4.007335261529810	0.4897362325985810
2.0	4.534192485682100	0.49462141854510300
2.25	5.76185759925369	0.5405394898833090
2.5	6.41954394976063	0.4523449367708450
2.75	6.023919923933230	0.48898047501417500
3.0	7.443742735575060	0.5342659697011440
3.25	7.870686775946450	0.4730380798563350
3.5	8.022587710435350	0.4572104512757930
3.75	9.395420889477870	0.49902577926821200
4.0	9.948288024421440	0.530527413902035
4.25	9.986332770533490	0.48969676619036700
4.5	9.694674681091840	0.5443240285483060
4.75	10.519384785632700	0.47616567472190800

Fitting In Action!

Curve-Fit with Python



```
import numpy as np
from scipy.optimize import curve_fit
from scipy.special import gammaincc

func = lambda x, a0, a1 : a0 + a1 * x

x, y, err = np.loadtxt('line.data',
                        unpack = True)

NDF = x.size - 2

popt, pcov = curve_fit(func, x, y,
                        sigma = err,
                        absolute_sigma = True)

perr = np.sqrt(np.diag(pcov))

chi2 = np.sum((y - func(x,*popt))**2 / err**2)
Q = gammaincc(0.5 * NDF, 0.5 * chi2)
```

Curve-Fit with Gnuplot

```
gnuplot> f(x) = a + b * x
gnuplot> fit f(x) "line.data" using 1:2:3 yerrors via a, b
```

...

!!!

```
degrees of freedom      (FIT_NDF)                : 18
rms of residuals        (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.967717
variance of residuals (reduced chisquare) = WSSR/ndf : 0.936476
p-value of the Chisq distribution (FIT_P)           : 0.532987
```

Final set of parameters

=====

```
a      = 0.838641
b      = 2.09749
```

Asymptotic Standard Error

=====

```
+/- 0.2066      (24.63%)
+/- 0.07469     (3.561%)
```

Different from the previous ones!

0.2134

0.0772

correlation matrix of the fit parameters:

```
      a      b
a      1.000
b     -0.856  1.000
```

Which ones are the right ones?

Chi-Squared

$$\chi^2 = \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2$$

- Assume the distribution of the deviation is “**Gaussian**” and uncorrelated.
- Assume $f(x)$ is *exact*.
- Then, χ^2 = sum of N squares of Gaussian random variables (normalized).
- It may **not** work well if your data have “**outliers**.”
- There are $(N - M)$ independent variables in the minimization.

$$N_{\text{DOF}} = N - M \quad (\text{Degrees of Freedom})$$

Why do you assume Gaussian?

If y_i is sampled from a parent Gaussian distribution with mean $f(x_i) = a_0 + a_1x$ and variance σ^2 ,

(Likelihood of y_i)

$$L_i = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(y_i - f(x_i))^2}{2\sigma^2} \right].$$

If the data points are uncorrelated,

(Likelihood of $\{y_1, \dots, y_N\}$)

$$L = \prod_{i=1}^N L_i \propto \exp \left[-\frac{1}{2} \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma} \right)^2 \right] = \exp \left[-\frac{1}{2} \chi^2 \right].$$

Maximum Likelihood



Minimizing Chi-Squared

Central Limit Theorem

- Sum of many random variables \rightarrow Gaussian distribution
- Independent, identically distributed random variables, of course.
- Finite average and variance

$$X = \sum_{i=1}^N x_i$$

$$\mu = \langle x \rangle$$

$$\sigma^2 = \langle x^2 \rangle - \langle x \rangle^2$$

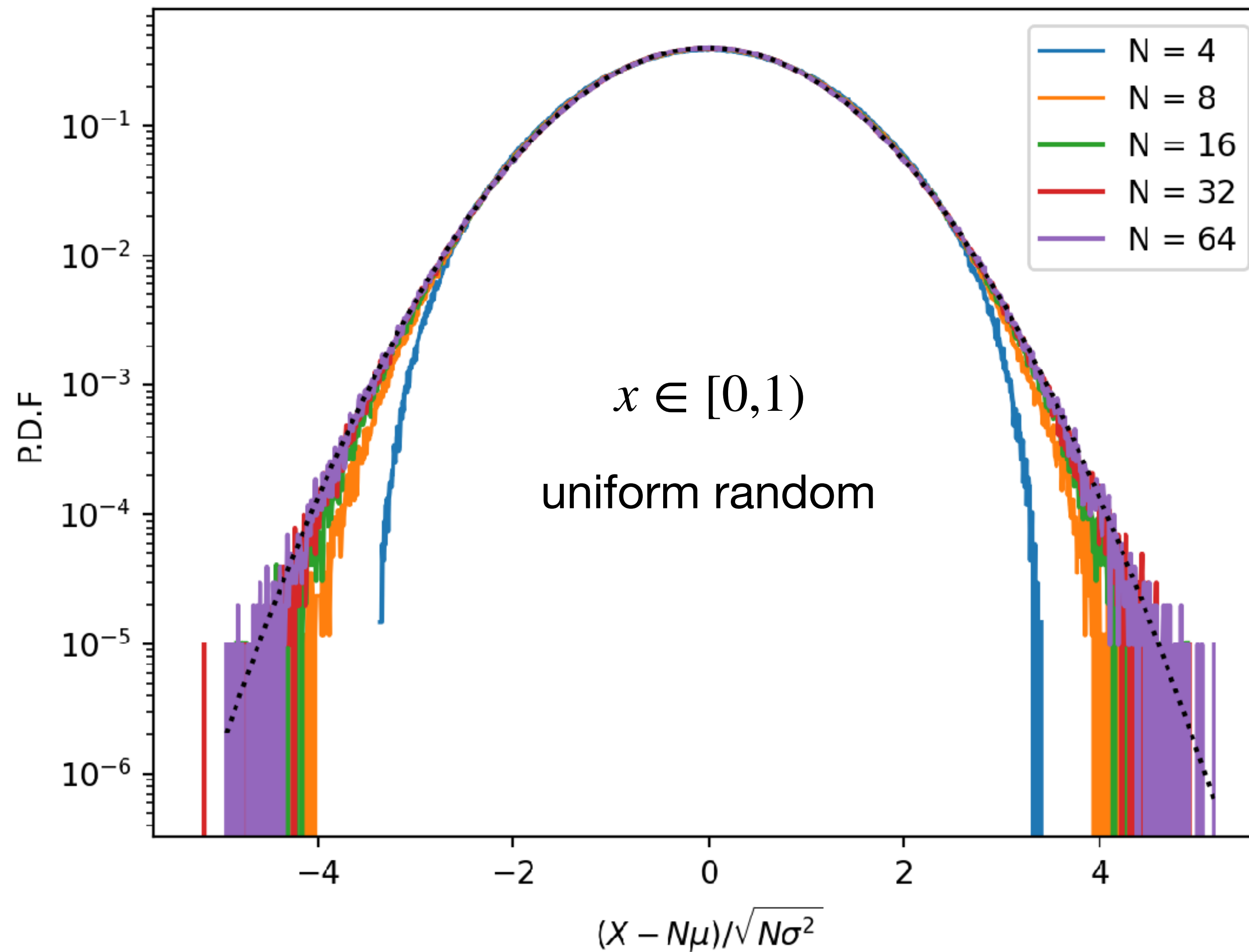
$$N \rightarrow \infty$$



$$P_N(X) = \frac{1}{\sqrt{2\pi N\sigma^2}} \exp \left[-\frac{(X - N\mu)^2}{2N\sigma^2} \right]$$

$$\langle X \rangle = N\mu \quad \langle X^2 \rangle - \langle X \rangle^2 = N\sigma^2$$

Central Limit Theorem



Linear model

The simplest case (M=2) : $f(x) = a_0 + a_1x$

$$\chi^2(a_0, a_1) = \sum_{i=1}^N \left(\frac{y_i - a_0 - a_1x_i}{\sigma_i} \right)^2 \xrightarrow{\text{Minimization}} \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} v_0 \\ v_1 \end{pmatrix}$$

They are Gaussian if the data noise is Gaussian!

$$a_0 = \frac{U_{11}v_0 - U_{01}v_1}{U_{00}U_{11} - U_{01}^2}$$

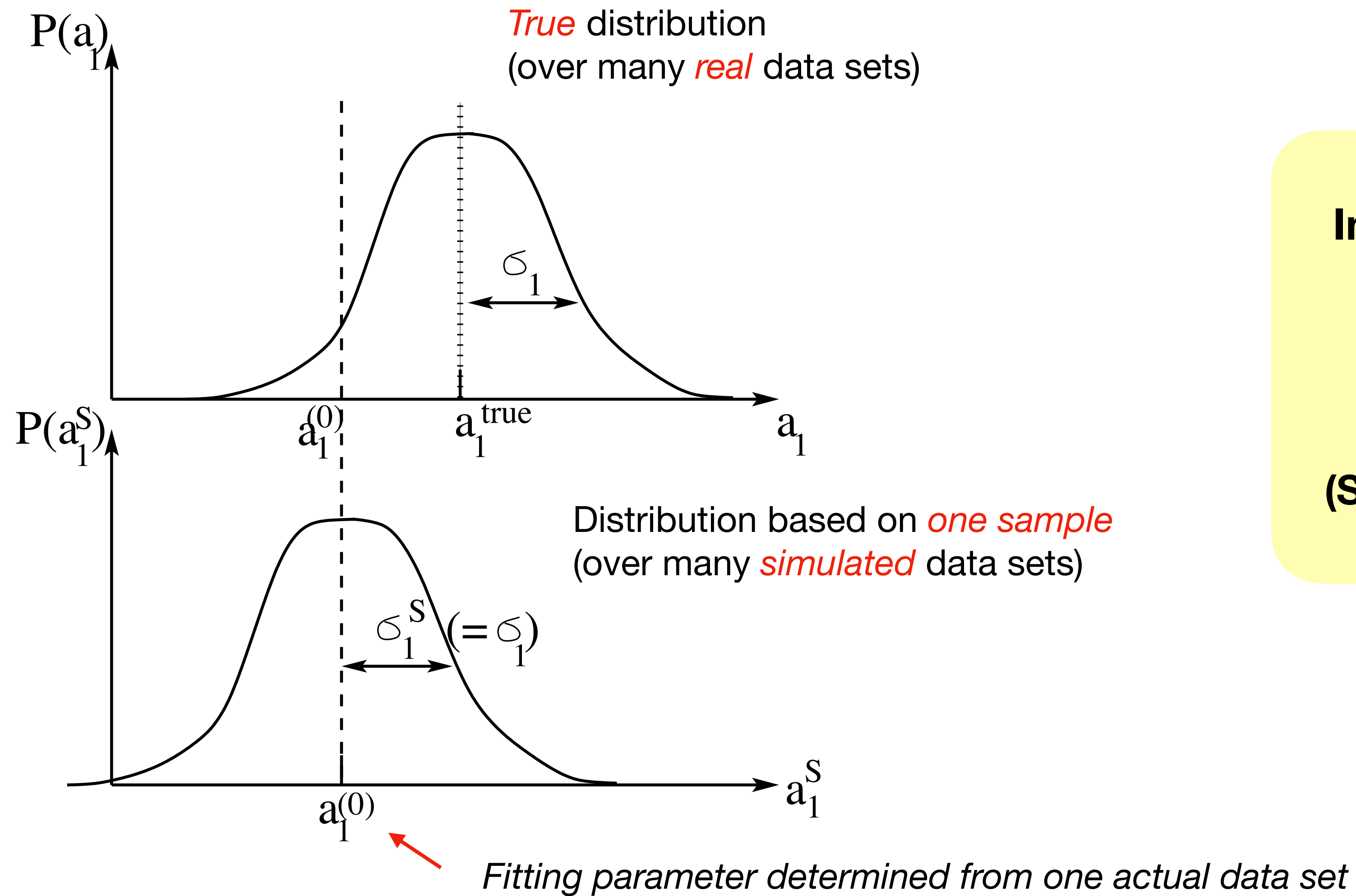
$$a_1 = \frac{-U_{01}v_0 + U_{00}v_1}{U_{00}U_{11} - U_{01}^2}$$

$$U_{\alpha\beta} = \sum_{i=1}^N \frac{x_i^{\alpha+\beta}}{\sigma_i^2} \quad v_\alpha = \sum_{i=1}^N \frac{y_i x_i^\alpha}{\sigma_i^2}$$

(symmetric, constant) (Sum of sample values)

U^{-1} : covariance matrix

Error estimates



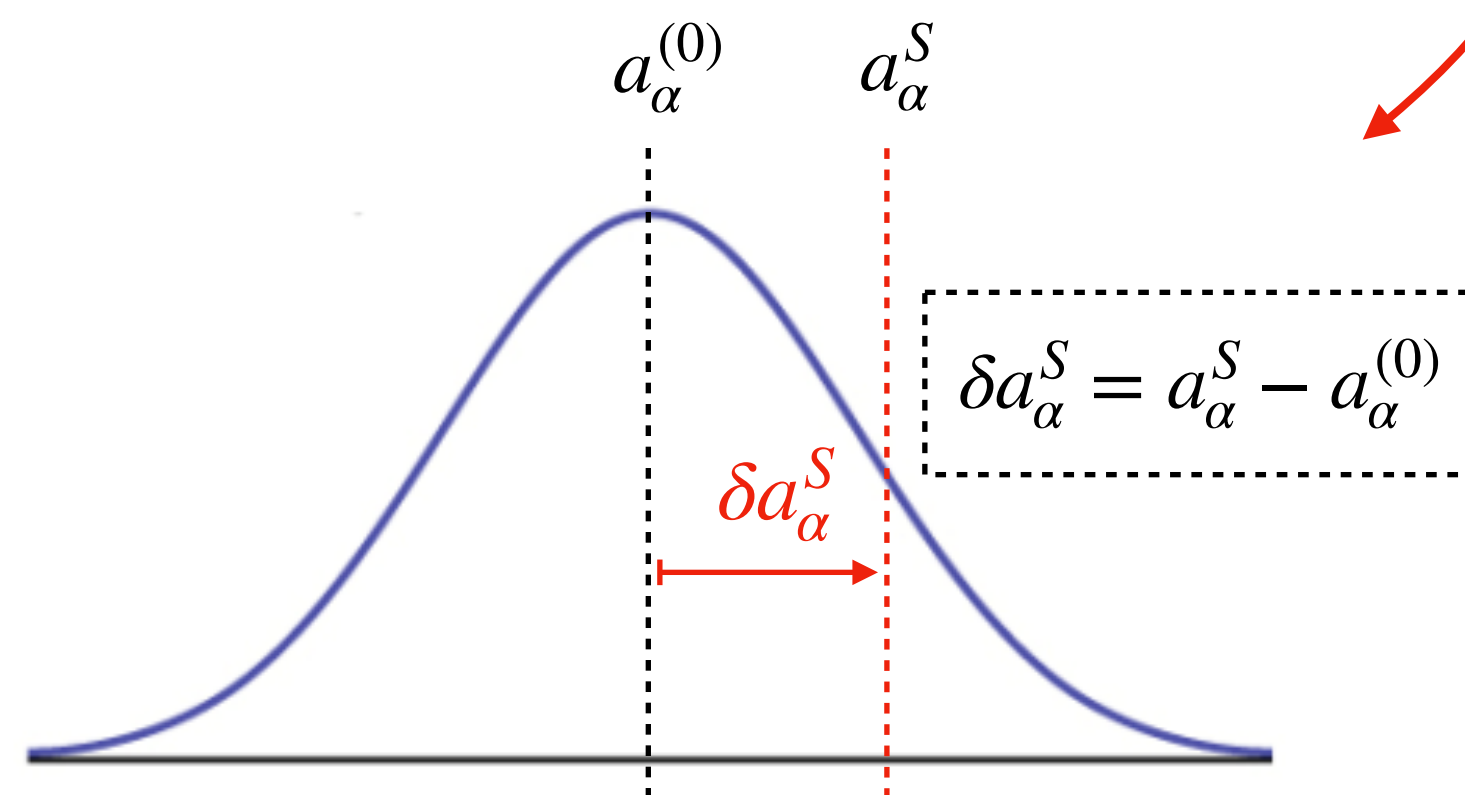
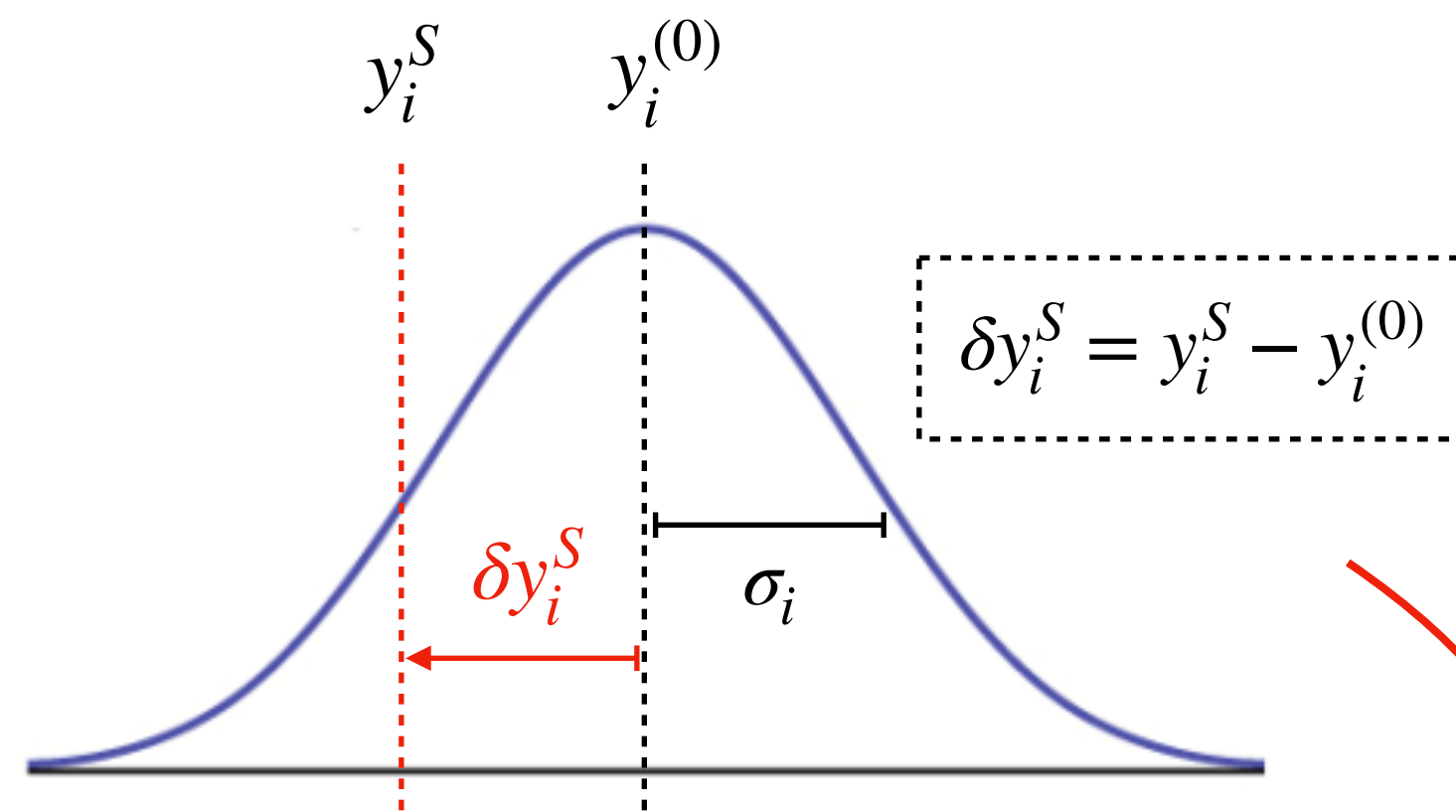
In the linear model,

$$\sigma_1^S = \sigma_1$$

(Simulated error bar) = (True error bar)

Error estimate

Simulated distribution centered at $y_i^{(0)}$



$$a_\alpha \equiv a_\alpha(\{y_i\})$$

$$\delta a_\alpha^S = \sum_{i=1}^N \frac{\partial a_\alpha}{\partial y_i} \delta y_i^S$$

**uncorrelated δy*

$$\rightarrow (\sigma_\alpha^S)^2 \equiv \langle (\delta a_\alpha^S)^2 \rangle = \sum_{i=1}^N \sigma_i^2 \left(\frac{\partial a_\alpha}{\partial y_i} \right)^2$$

$$(\sigma_\alpha^S)^2 = \sum_{\beta, \gamma} [U^{-1}]_{\alpha\beta} [U^{-1}]_{\alpha\gamma} \left[\sum_{i=1}^N \frac{x_i^{\beta+\gamma}}{\sigma_i^2} \right] = [U^{-1}]_{\alpha\alpha}$$

Error estimate of the fitting parameter

$$(\sigma_\alpha^S)^2 = [U^{-1}]_{\alpha\alpha}$$

i.e. it's independent of "S": $\sigma_\alpha^S = \sigma_\alpha$

Covariance matrix

Correlation between fluctuations of fitting parameters

$$\text{Cov}(\alpha, \beta) = \langle \delta a_\alpha \delta a_\beta \rangle = \sum_{i,j} \frac{\partial a_\alpha}{\partial y_i} \frac{\partial a_\beta}{\partial y_j} \langle \delta y_i \delta y_j \rangle = \sum_i \sigma_i^2 \frac{\partial a_\alpha}{\partial y_i} \frac{\partial a_\beta}{\partial y_i} = \sum_{\mu,\nu} [\mathbf{U}^{-1}]_{\alpha\mu} [\mathbf{U}^{-1}]_{\beta\nu} \mathbf{U}_{\nu\mu} = [\mathbf{U}^{-1}]_{\alpha\beta}$$

Correlation coefficient

$$r_{\alpha\beta} = \frac{\text{Cov}(\alpha, \beta)}{\sigma_\alpha \sigma_\beta} = \frac{[\mathbf{U}^{-1}]_{\alpha\beta}}{\sigma_\alpha \sigma_\beta}$$

$$a_\alpha = \sum_\beta [\mathbf{U}^{-1}]_{\alpha\beta} \sum_{i=1}^N \frac{y_i x_i^\beta}{\sigma_i^2}$$
$$U_{\alpha\beta} = \sum_{i=1}^N \frac{x_i^{\alpha+\beta}}{\sigma_i^2}$$

Quality of the fit

Null hypothesis (Chi-squared distribution)

$$P^{(m)}(X) = \frac{1}{2^{m/2}\Gamma(m/2)} X^{(m/2)-1} e^{-X/2}$$

$$X \equiv \chi^2 = \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2$$

$$m = N - M \equiv N_{\text{DOF}}$$

1. Generate a sample of random variables $\{y_0, \dots, y_N\}$ from their parent Gaussian distributions.
2. Minimize χ^2 .
3. Repeat [1-2] to generate many χ^2 samples
4. Now, you have the probability distribution of χ^2 under *null* hypothesis.

Derivation of χ^2 -distribution

Residual (Gaussian random variable, $\mathcal{N}(0,1)$) $\epsilon_i = \frac{y_i - a_0 - a_1 x_i}{\sigma_i}$

Two equation (minimizing χ^2) $\sum_{i=1}^N \frac{1}{\sigma_i} \epsilon_i = 0 \quad \sum_{i=1}^N \frac{x_i}{\sigma_i} \epsilon_i = 0 \longrightarrow \mathbf{M}\boldsymbol{\epsilon} = 0$

Number of independent random variables

$$m = N - 2 \equiv N_{\text{DOF}}$$

$$\begin{array}{c} \downarrow \text{(SVD)} \\ \mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \end{array}$$

Independent random variables spanning the null space ($\mathbf{M}\boldsymbol{\epsilon} = 0$)

$$\mathbf{z} = \mathbf{V}^T \boldsymbol{\epsilon}$$

($\mathcal{N}(0,1)$ $\mathbf{N} \rightarrow \mathbf{N}_{\text{DOF}}$ random variables)

$$\|\mathbf{z}\|_2^2 = \|\boldsymbol{\epsilon}\|_2^2 = \chi^2$$

(orthogonal transformation)

Derivation of χ^2 -distribution

Surface of a unit m -dim. sphere

Joint probability distribution of (z_0, z_1, \dots, z_m)

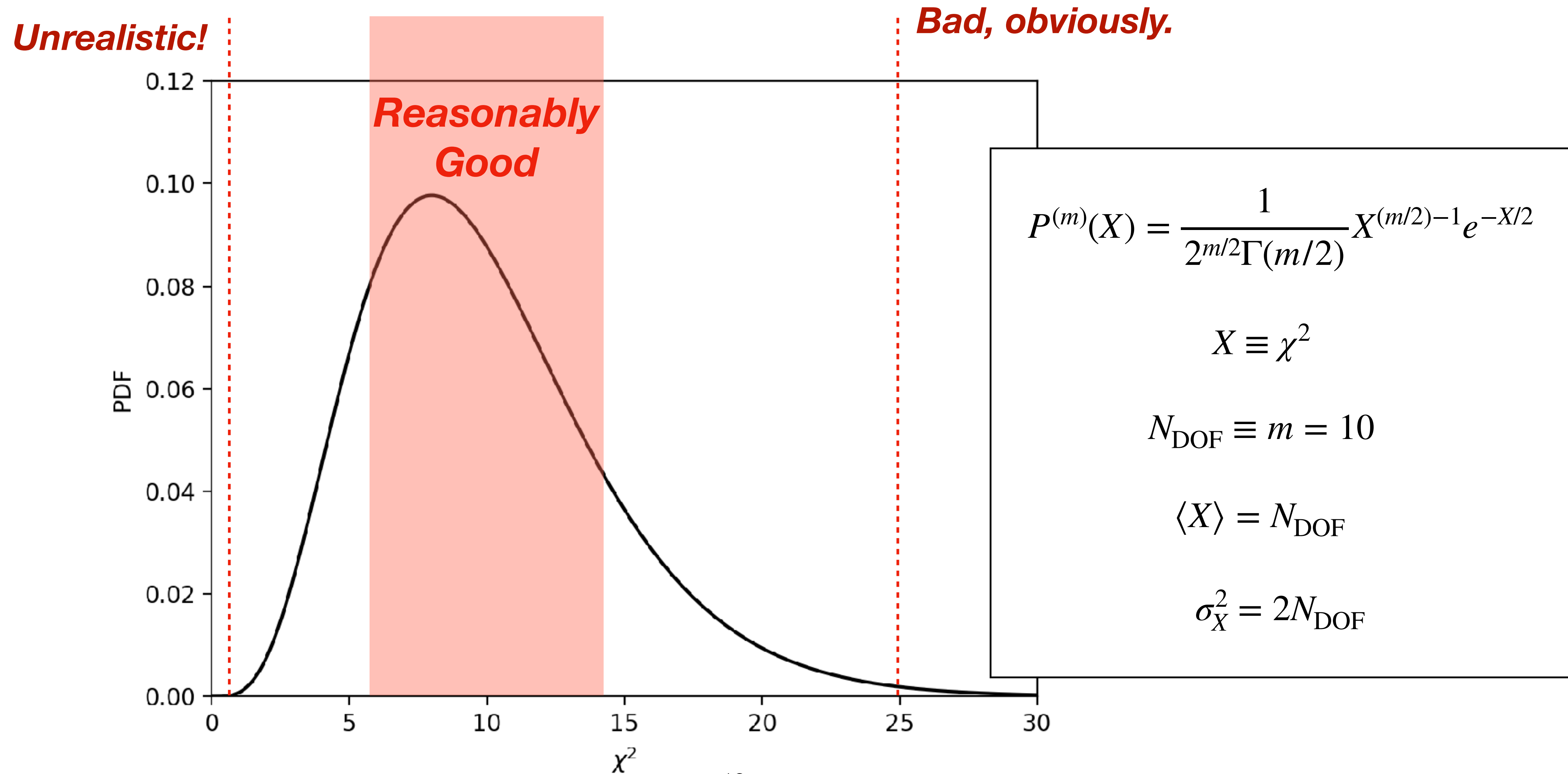
$$P(\mathbf{z})d\mathbf{z} = \prod_{i=1}^m P(z_i) dz_i = \prod_{i=1}^m \frac{dz_i}{\sqrt{2\pi}} e^{-z_i^2/2} \xrightarrow{r^2 = \sum_i z_i^2} P^{(m)}(r) dr = \frac{S_m}{(2\pi)^{m/2}} r^{m-1} e^{-r^2/2} dr$$

$$S_m = \frac{2\pi^{m/2}}{\Gamma(m/2)}$$

Chi-Squared distribution ($X \equiv \chi^2$)

$$P^{(m)}(X)dX = P^{(m)}(r)dr \longrightarrow P^{(m)}(X) = \frac{1}{2^{m/2}\Gamma(m/2)} X^{(m/2)-1} e^{-X/2}$$

$$1 - s\sqrt{\frac{2}{N_{\text{DOF}}}} \lesssim \frac{\chi^2}{N_{\text{DOF}}} \lesssim 1 + s\sqrt{\frac{2}{N_{\text{DOF}}}}$$



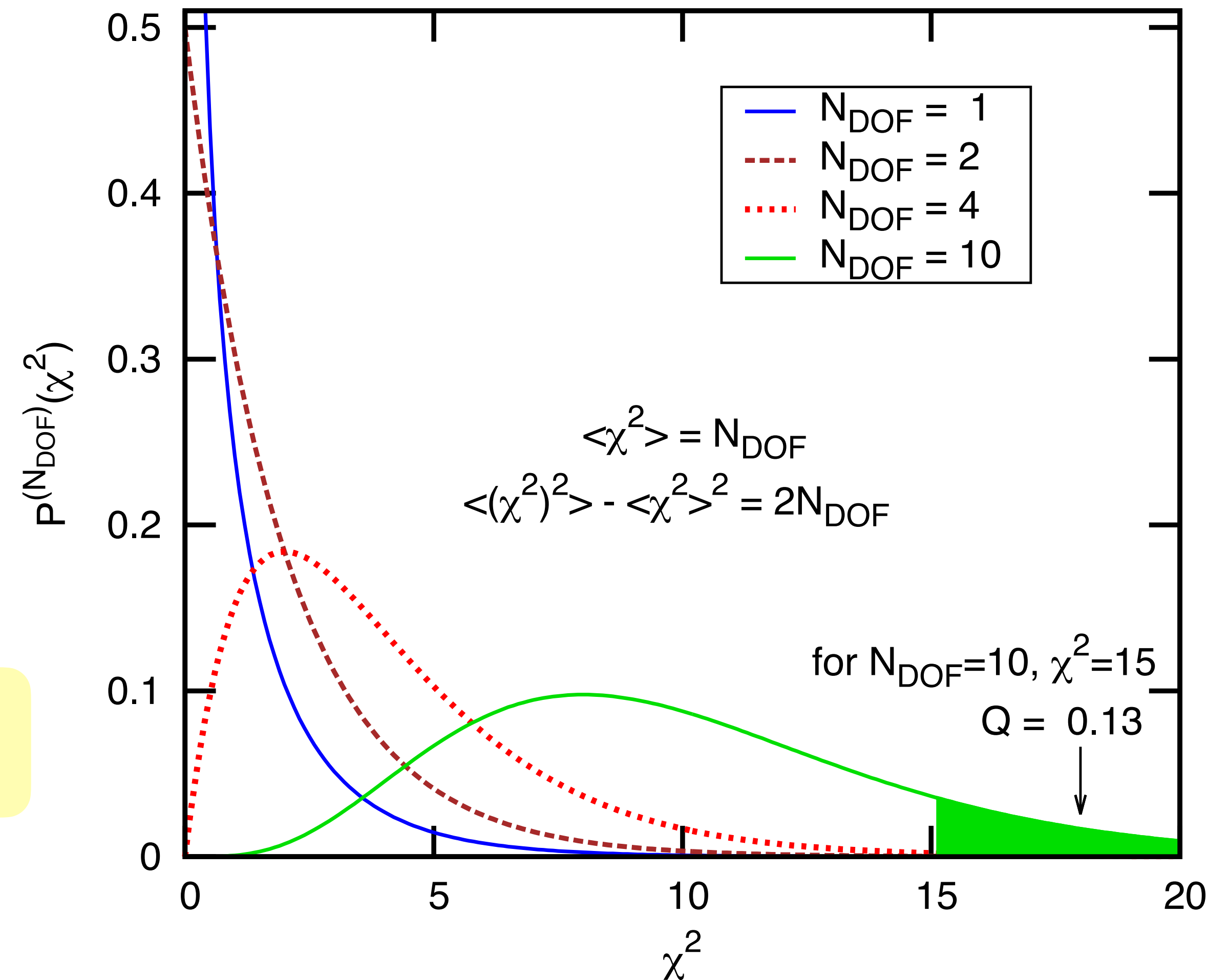
Goodness-of-Fit

- **Reduced Chi-Squared Statistic**

$$\frac{\chi^2}{N_{\text{DOF}}} \sim 1$$

- **P-value:** $Q(\chi^2) = P(X > \chi^2)$

$$Q(\chi^2) = \frac{1}{\Gamma(N_{\text{DOF}}/2)} \int_{\frac{\chi^2}{2}}^{\infty} y^{\frac{N_{\text{DOF}}}{2}-1} e^{-y} dy \sim 0.5$$



Asymptotic standard error

- What if you do not have any error bars available on your data points?
- Still, the chi-squared can make error estimates on the fit parameters.
- Assumption 1. All data points have **equal** error bars. i.e. $\sigma_i = \sigma_{\text{ass}}$.
- Assumption 2. Your data points fit **reasonably** well to your model.

→

$$1 = \frac{\chi_{\text{ass}}^2}{N_{\text{DOF}}} = \frac{1}{N_{\text{DOF}}} \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma_{\text{ass}}} \right)^2$$

Asymptotic standard error

How to get the assumed error :

$$\sigma_{\text{ass}}^2 = \frac{1}{N_{\text{DOF}}} \sum_{i=1}^N (y_i - f(x_i))^2$$

$$1 = \frac{\chi_{\text{ass}}^2}{N_{\text{DOF}}} = \frac{1}{N_{\text{DOF}}} \sum_{i=1}^N \left(\frac{y_i - f(x_i)}{\sigma_{\text{ass}}} \right)^2$$

Just do the least-square fitting with $\sigma_i = 1$.

$$\chi^2 = \sum_{i=1}^N (y_i - f(x_i))^2$$

Minimize!

$$\sigma_{\text{ass}}^2 = \frac{\chi_{\text{min}}^2}{N_{\text{DOF}}}$$

$$[\mathbf{U}^{-1}]_{\alpha\alpha} \longrightarrow$$

Asymptotic Standard Error

$$\sigma_{\alpha}^2 = [\mathbf{U}^{-1}]_{\alpha\alpha} \sigma_{\text{ass}}^2$$

Coming back to Gnuplot ...

```
gnuplot> f(x) = a + b * x
gnuplot> fit f(x) "line.data" using 1:2:3 yerrors via a, b
```

...

!!!

```
degrees of freedom      (FIT_NDF)                : 18
rms of residuals        (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.967717
variance of residuals (reduced chisquare) = WSSR/ndf : 0.936476
p-value of the Chisq distribution (FIT_P)           : 0.532987
```

Final set of parameters

=====

```
a      = 0.838641
b      = 2.09749
```

Asymptotic Standard Error

=====

```
+/- 0.2066      (24.63%)
+/- 0.07469     (3.561%)
```

Different from the previous ones!

0.2134

0.0772

correlation matrix of the fit parameters:

```
      a      b
a      1.000
b     -0.856  1.000
```

Which ones are the right ones?

Coming back to Gnuplot ...

```
degrees of freedom      (FIT_NDF)                : 18
rms of residuals        (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.967717
variance of residuals (reduced chisquare) = WSSR/ndf : 0.936476
p-value of the Chisq distribution (FIT_P)           : 0.532987
```

Final set of parameters	Asymptotic Standard Error
=====	=====
a = 0.838641	+/- 0.2066 (24.63%)
b = 2.09749	+/- 0.07469 (3.561%)

Asymptotic Standard Error

$$\sigma_{\alpha}^2 = [\mathbf{U}^{-1}]_{\alpha\alpha} \sigma_{\text{ass}}^2$$

*Gnuplot computes it
with weights properly.*

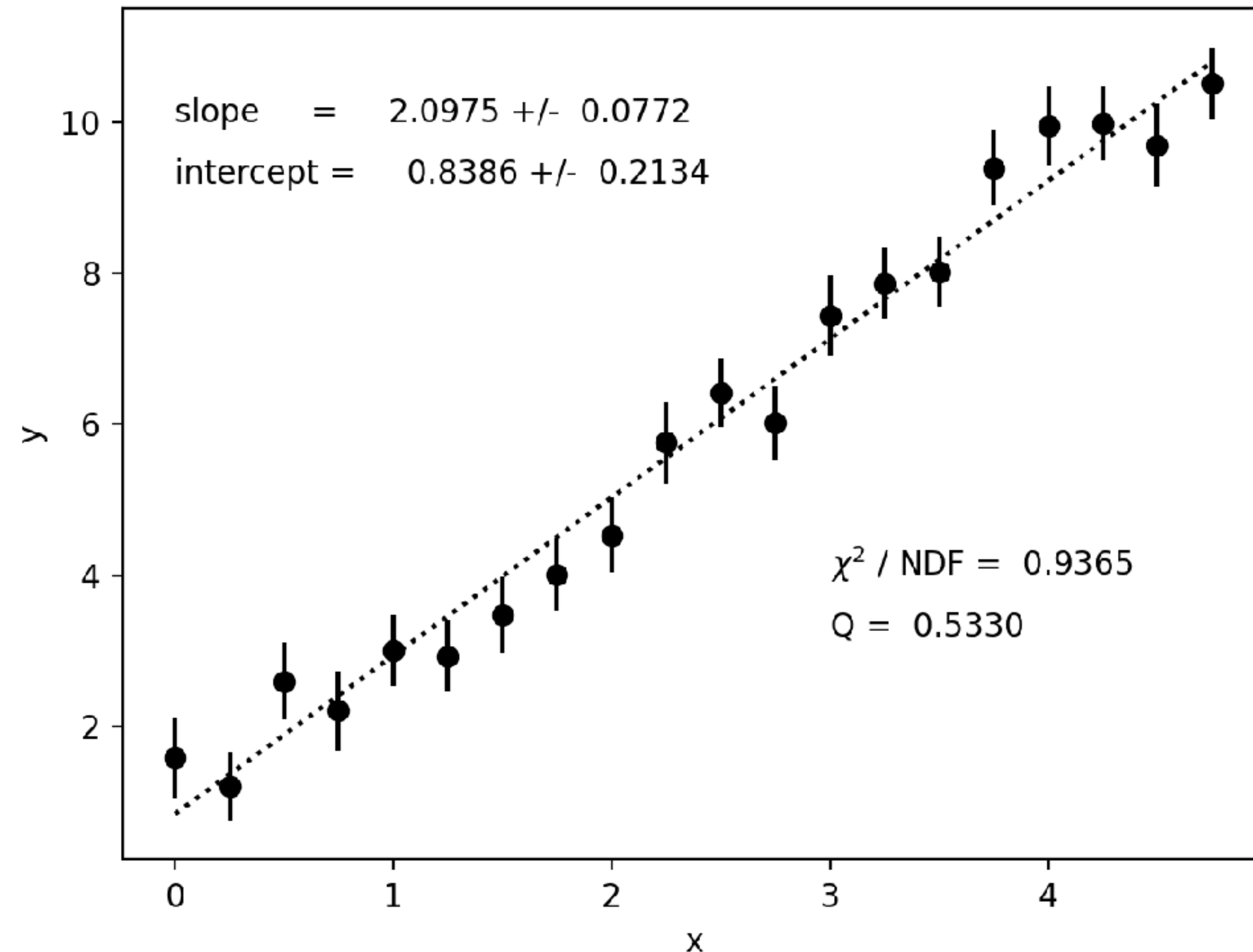
Then, it shouldn't appear.

Correct error bars :

0.2134 (scipy) = 0.2066 / FIT_STDFIT

0.0772 (scipy) = 0.07469 / FIT_STDFIT

Caution with scipy's curve_fit()



```
import numpy as np
from scipy.optimize import curve_fit
from scipy.special import gammaincc

func = lambda x, a0, a1 : a0 + a1 * x

x, y, err = np.loadtxt('line.data',
                        unpack = True)

NDF = x.size - 2

popt, pcov = curve_fit(func, x, y,
                        sigma = err,
                        absolute_sigma = True)

perr = np.sqrt(np.diag(pcov))

chi2 = np.sum((y - func(x,*popt))**2 / err**2)
Q = gammaincc(0.5 * NDF, 0.5 * chi2)
```

Caution with scipy's curve_fit()

```
def curve_fit(f, xdata, ydata, p0=None, sigma=None, absolute_sigma=False,
              check_finite=True, bounds=(-np.inf, np.inf), method=None,
              jac=None, **kwargs):
```

`absolute_sigma` : bool, optional

!!!!!!

If True, ``sigma`` is used in an absolute sense and the estimated parameter covariance ``pcov`` reflects these absolute values.

**Set “sigma” with
“absolute_sigma = True” option**

If False (default), only the relative magnitudes of the ``sigma`` values matter.

The returned parameter covariance matrix ``pcov`` is based on scaling

``sigma`` by a constant factor. This constant is set by demanding that the

reduced ``chisq`` for the optimal parameters ``popt`` when using the

`*scaled*` ``sigma`` equals unity. In other words, ``sigma`` is scaled to

match the sample variance of the residuals after the fit. Default is False.

Mathematically,

``pcov(absolute_sigma=False) = pcov(absolute_sigma=True) * chisq(popt)/(M-N)``

Generalized linear model

Linear model does not necessarily mean a straight line.

$$f(x) = \sum_{\alpha=1}^M a_{\alpha} X_{\alpha}(x) \quad \text{with basis functions} \quad X_1(x), X_2(x), \dots, X_M(x) \quad (\text{ex. polynomial})$$

↓ *Minimize*

$$\chi^2 = \sum_{i=1}^N \left(\frac{y_i - \sum_{\alpha=1}^M a_{\alpha} X_{\alpha}(x_i)}{\sigma_i} \right)^2$$

The same linear equation to solve

$$\mathbf{U}\mathbf{a} = \mathbf{v} \quad U_{\alpha\beta} = \sum_{i=1}^N \frac{1}{\sigma_i^2} X_{\alpha}(x_i) X_{\beta}(x_i) \quad v_{\alpha} = \sum_{i=1}^N \frac{y_i}{\sigma_i^2} X_{\alpha}(x_i)$$

Power-law fitting

It can be converted into a linear model.

$$y = a_0 x^{a_1}$$



$$\log y = \log a_0 + a_1 \log x$$



$$Y = a'_0 + a_1 X$$

What about the uncertainties? They are not exactly Gaussian.

→ It does not matter if the uncertainties are *sufficiently small*.

$$\log(y + \sigma) = \log \left[y \left(1 + \frac{\sigma}{y} \right) \right] \simeq \log y + \frac{\sigma}{y} \quad \text{when}$$

$$\frac{\sigma}{y} \ll 1$$

Nonlinear model

- One can still try the least square fitting with a general function.
- One may be able to write an equation like $\mathbf{U}\mathbf{a} = \mathbf{v}$, however the curvature matrix \mathbf{U} now depends on the fit parameters \mathbf{a} . In the linear model, \mathbf{U} and \mathbf{v} does not depend on \mathbf{a} .
- Although, one still minimize χ^2 numerically by using the Levenberg-Marquardt method. See Numerical Recipes, if you're interested.
- The quantity $(\sigma_\alpha^S)^2 = [\mathbf{U}^{-1}]_{\alpha\alpha}$ can be evaluated at minimum χ^2 , but its validity as an error estimator depends on the spread of fitted parameters.

Confidence limit

- It estimates the error bar of a fitted parameter by setting a criterion for $\Delta\chi^2 \equiv \chi^2 - \chi_{\min}^2$, the deviation from minimum χ^2 .
- Joint probability distribution of fit parameters $\mathbf{a}^S = (a_1^S, a_2^S, \dots, a_M^S)$

See Appendix E of [arXiv:1210.3781](https://arxiv.org/abs/1210.3781).

$$P(\mathbf{a}^S) \propto \exp \left(-\frac{1}{2} \sum_{\alpha, \beta} \delta a_{\alpha}^S U_{\alpha\beta} \delta a_{\beta}^S \right) = \exp \left(-\frac{1}{2} \Delta\chi^2 \right)$$

$$\chi^2 \approx \chi_{\min}^2 + \frac{1}{2} \frac{\partial^2 \chi^2}{\partial a_{\alpha} \partial a_{\beta}} \bigg|_{\min} \delta a_{\alpha}^S \delta a_{\beta}^S \quad \leftarrow \quad \frac{1}{2} \frac{\partial^2 \chi^2}{\partial a_{\alpha} \partial a_{\beta}} \bigg|_{\min} = \sum_{i=1}^N \frac{1}{\sigma_i^2} X_{\alpha}(x_i) X_{\beta}(x_i) = U_{\alpha\beta} \quad (\text{effective linear model})$$

Confidence limit

$$P(\mathbf{a}^S) \propto \exp\left(-\frac{1}{2}\Delta\chi^2\right)$$

Does $\Delta\chi^2 = 1$ mean something particular?

So-call “one sigma”, 68% confidence limit.

(c.f “two sigma” [$\Delta\chi^2 = 4$] : 95% confidence limit)

$$\Delta\chi^2 = 1$$

Linear model

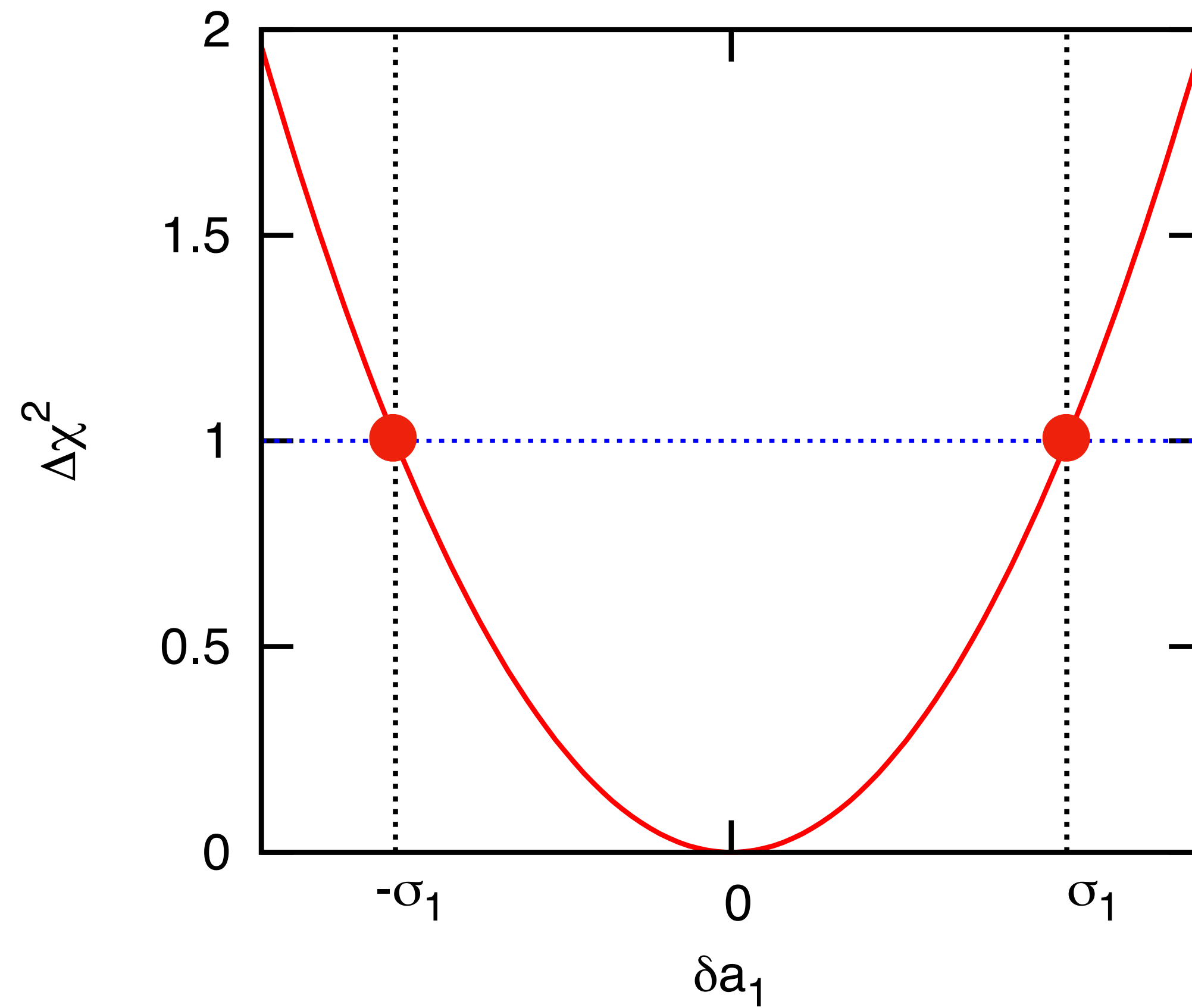
$$(\sigma_\alpha^S)^2 = [U^{-1}]_{\alpha\alpha}$$

Compute $P(a_1^S)$: fix a_1^S at a given value, adjust the others to minimize χ^2 .

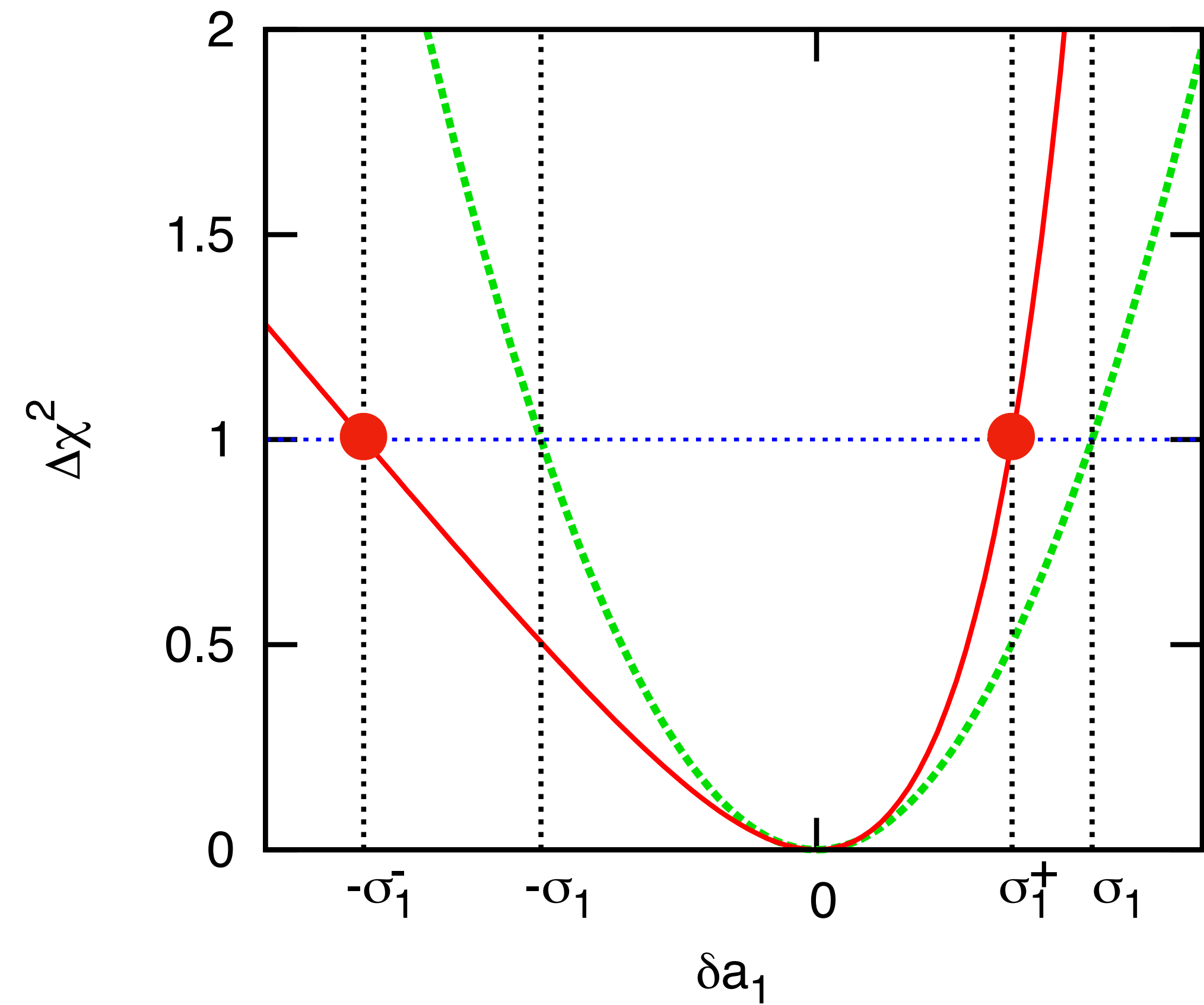
$$\frac{\partial\chi^2}{\partial a_\alpha} = 0 \quad (\alpha \neq 1) \quad \longrightarrow \quad \sum_{\beta=1}^M U_{\alpha\beta} \delta a_\beta^S = c \delta_{\alpha,1} \quad \longrightarrow \quad c = \frac{\delta a_1^S}{[U^{-1}]_{11}} \quad \longrightarrow \quad P(a_1^S) \propto \exp\left(-\frac{(\delta a_1^S)^2}{2[U^{-1}]_{11}}\right)$$

Confidence limit

Linear model




Nonlinear model (asymmetric error bars?)



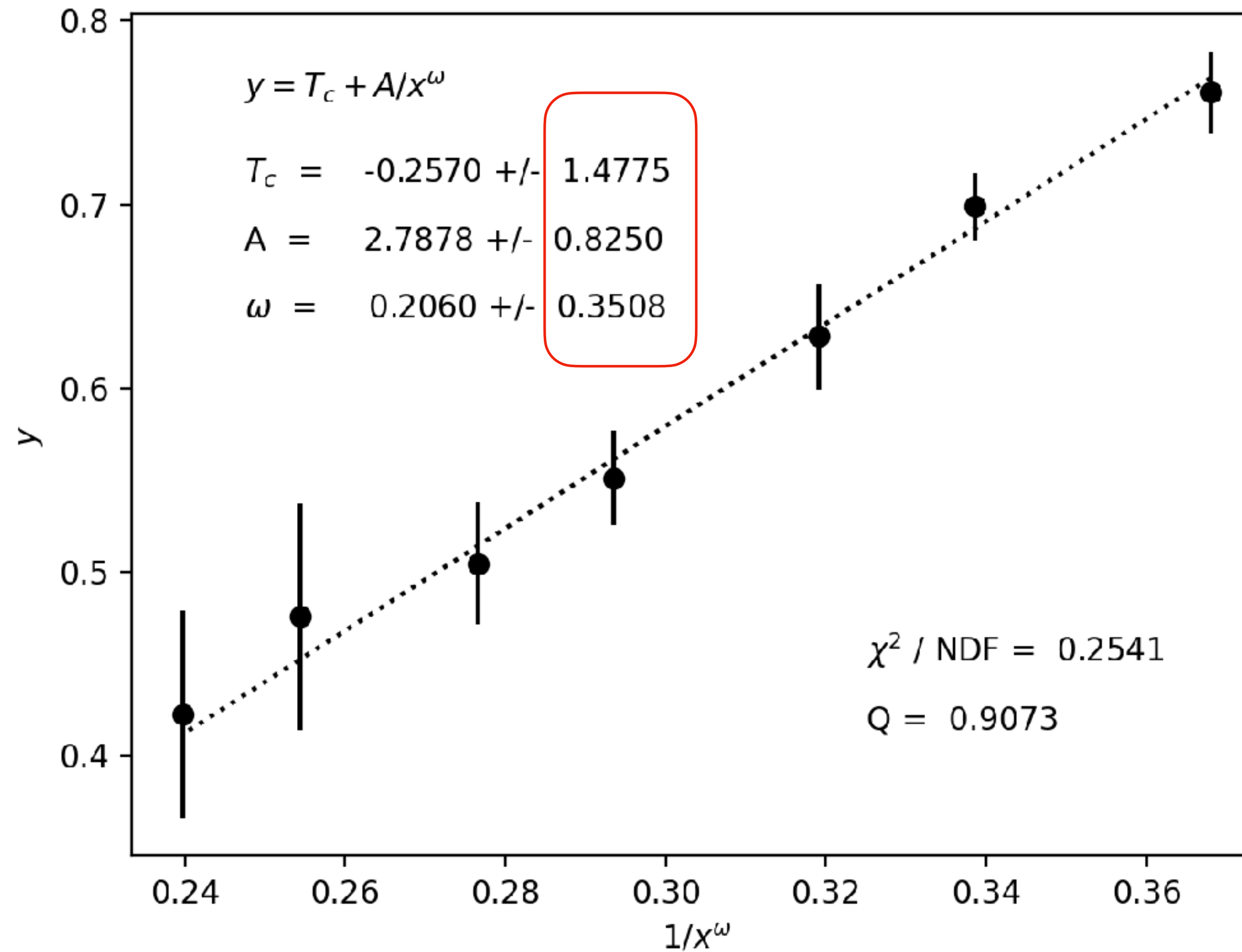
Example: $y = T_c + A / x^w$

x	y	error
128	0.760881	0.022157
192	0.698935	0.018346
256	0.628069	0.028605
384	0.550988	0.025836
512	0.504440	0.033009
768	0.475425	0.061739
1024	0.422427	0.056518

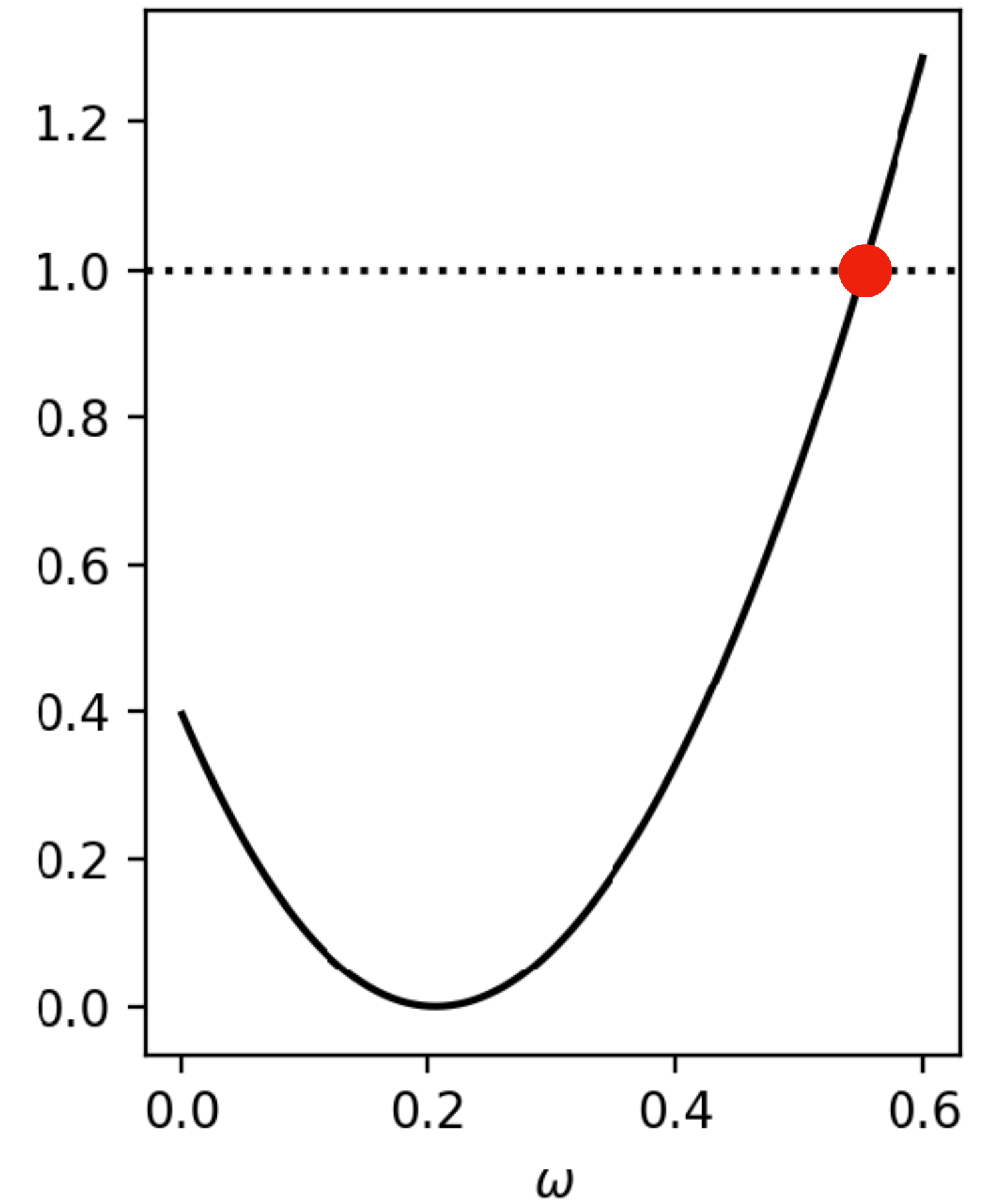
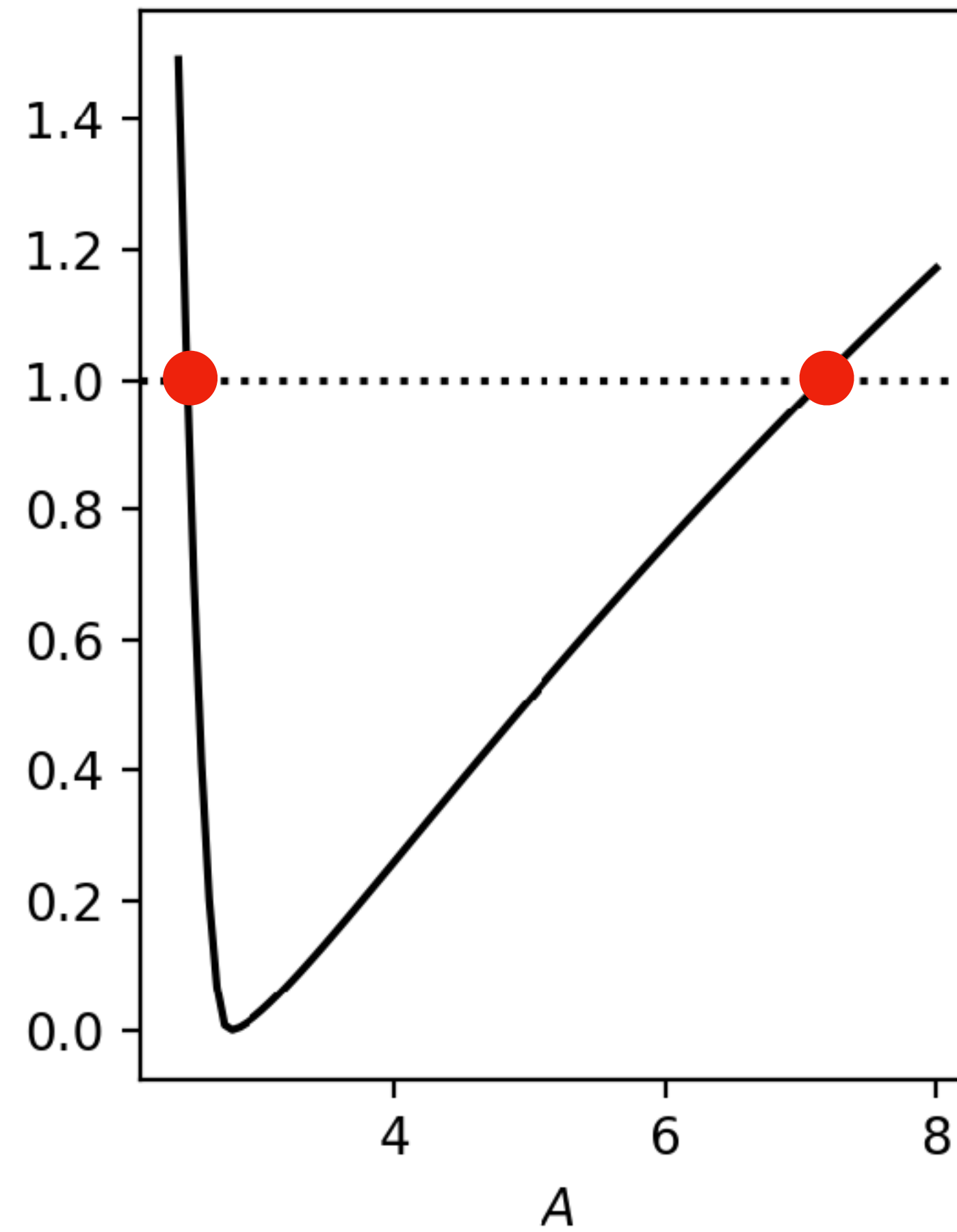
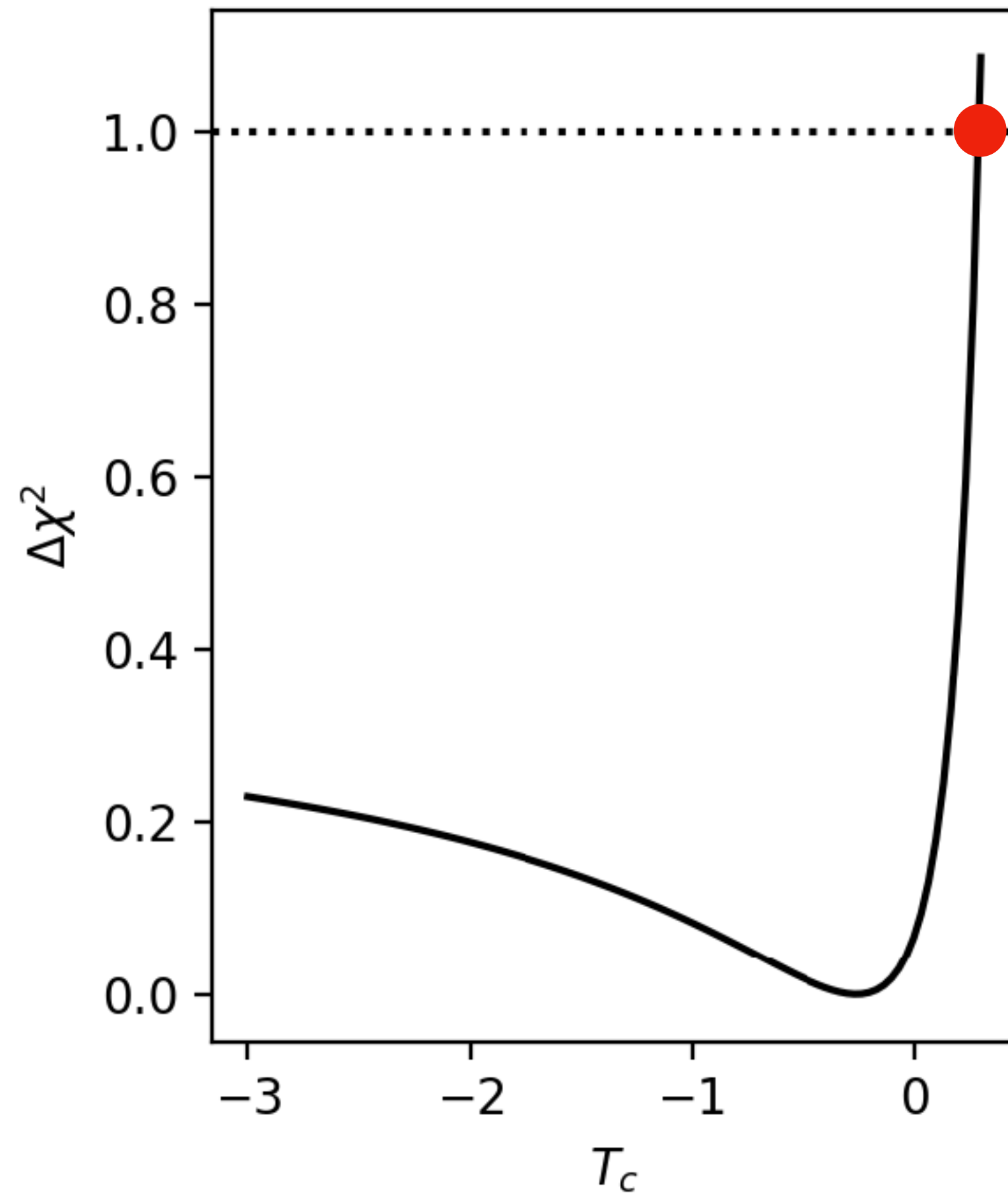
Least square fit with a nonlinear model


 $y = T_c + A/x^w$

Example: error bars with an effective linear model



Example: highly asymmetric error bars



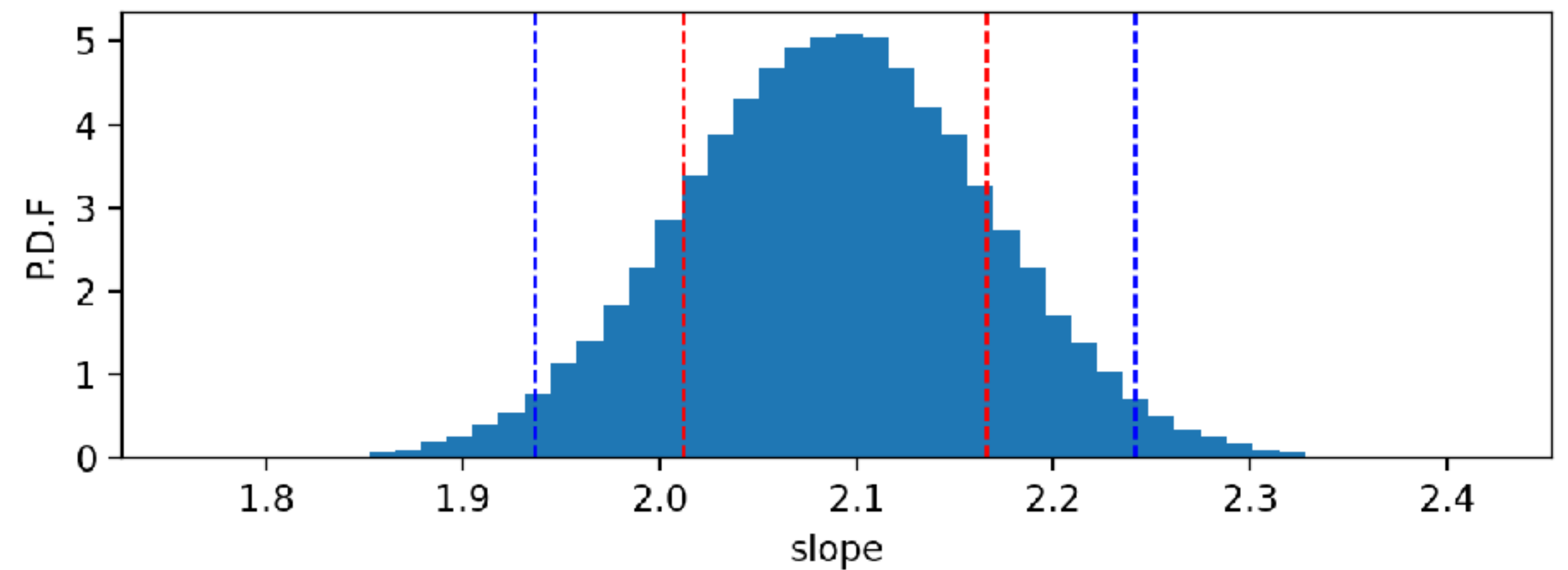
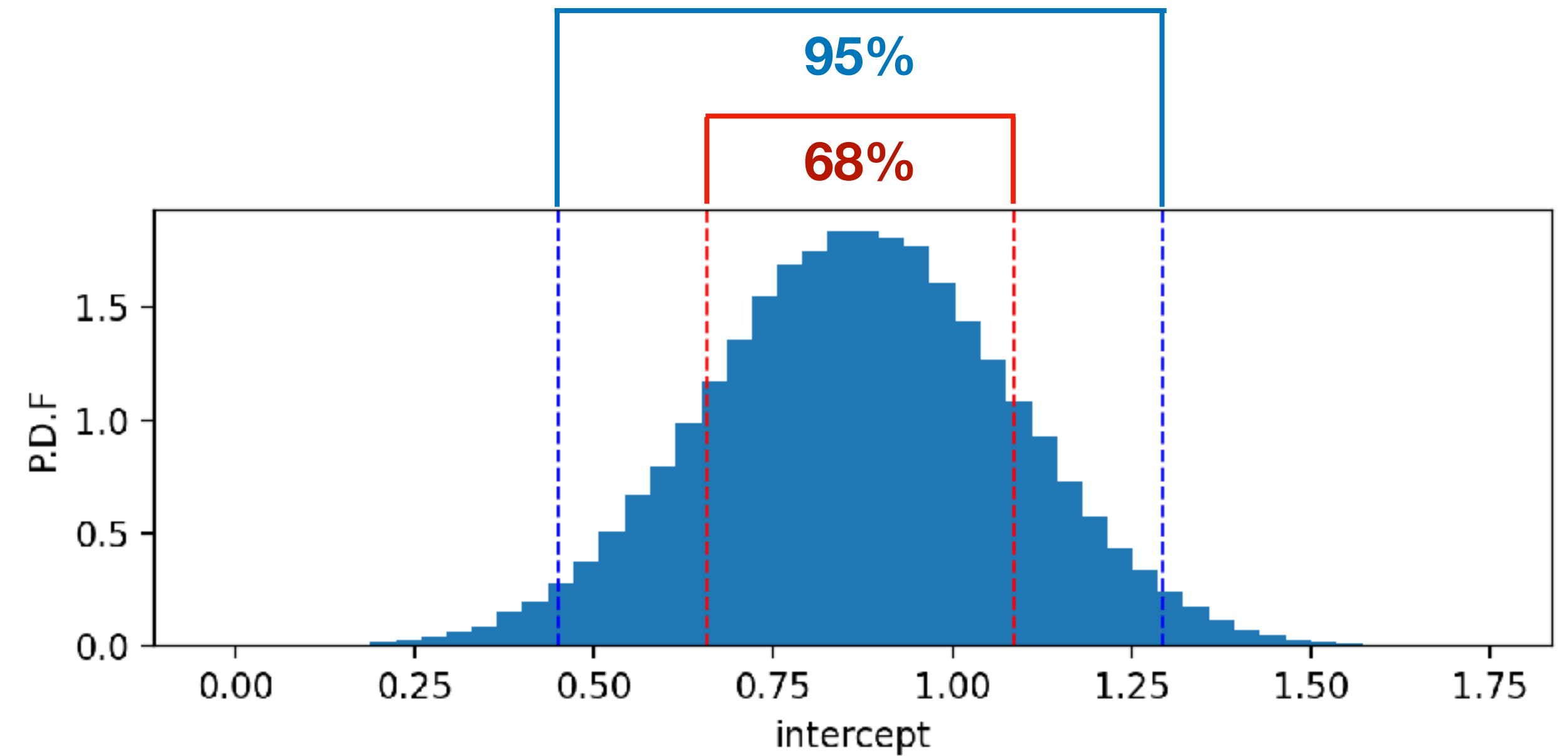
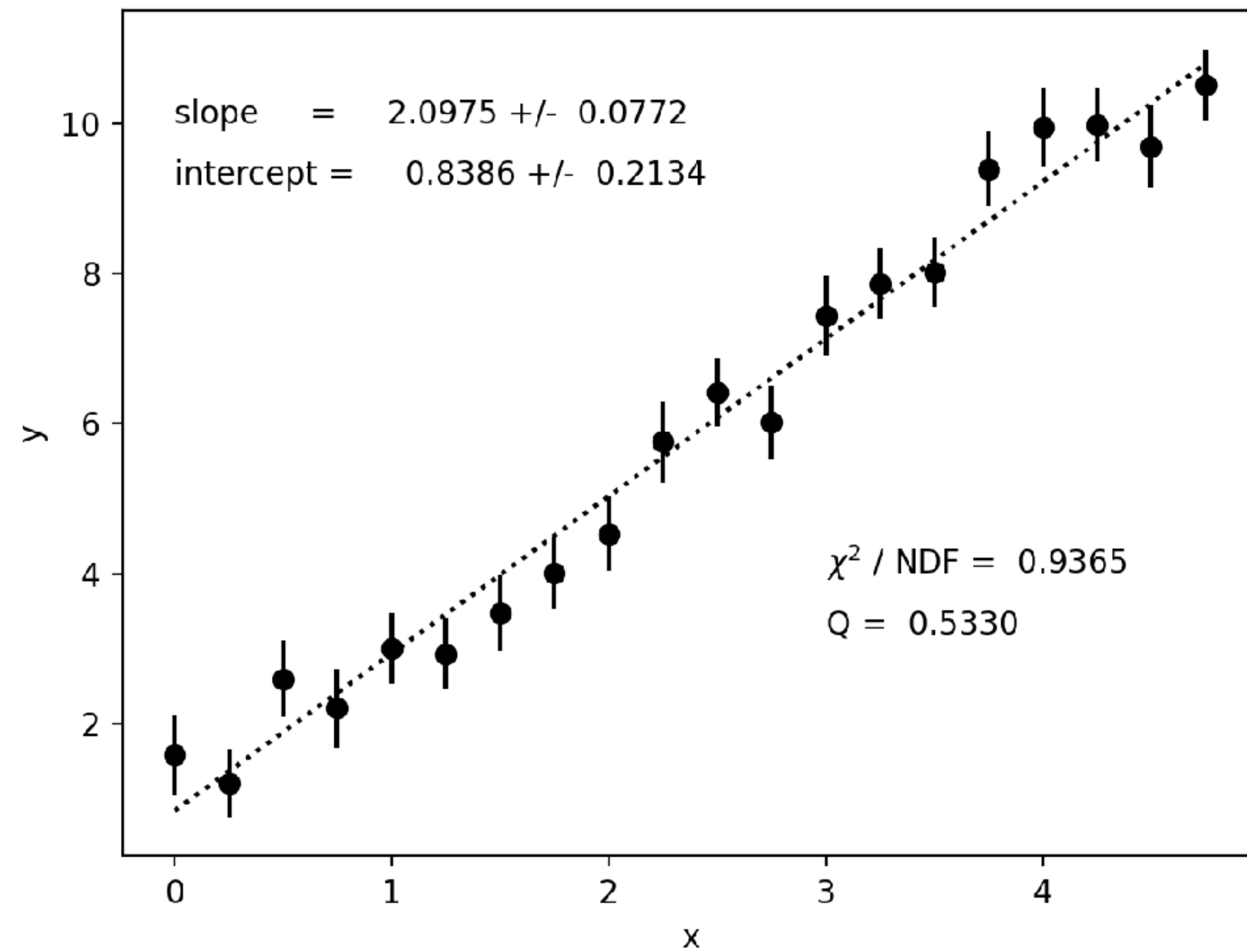
Confidence limit by resampling the data

- You have access to a raw data set, which may exhibit a *non-Gaussian* distribution, producing a data point (x_i, y_i) .
- If it is the case, “*bootstrap resampling*” gives you *better* error estimates.
- Make many “bootstrap” data sets. Find the fit parameters for each of them to get statistics. Plot the histogram, and find the confidence interval.
- Though you do not have access to raw data, you can still make a *synthetic (simulated) raw data*, assumed that the noises are Gaussian.

Bootstrap resampling

- For a raw data set $\{y_1^{(i)}, y_2^{(i)}, \dots, y_{N_i}^{(i)}\}$ that produces each data point $y_i = \bar{y}_j^{(i)}$, one can generate a “bootstrap” data set by randomly picking up N_i element from the raw data set with ***replacement***.
- Instead of one set of the original data points $\mathbf{y} = (y_1, y_2, \dots, y_N)$, one can consider a “bootstrap” data points $\mathbf{y}^* = (y_1^*, y_2^*, \dots, y_N^*)$ to make the least square fit to determine the fitting parameters $\mathbf{a}^* = (a_0^*, a_1^*, \dots, a_{M-1}^*)$.
- Generating many such “bootstrap” data points makes many different sets of the fitting parameters, allowing us to draw a probability distribution of $\mathbf{a}^* = (a_0^*, a_1^*, \dots, a_{M-1}^*)$.
- Choose the confidence interval, typically 95% (two-sigma), 68% (one-sigma), ...

Confidence interval



Summary

- Least-square fitting works with **Gaussian error bars** and **linear models**.
- Gnuplot: be careful with “asymptotic standard error.”
- Python: make sure to choose the right “absolute_sigma” option.
- It may work for non-Gaussian error bars and nonlinear models, but use it with caution when trying to estimate errors of fitting parameters.
- You may consider the bootstrap resampling if you have raw data sets or simulated data sets.