

# Week 8

## Issues in Non-linear Fitting

# Linear vs. Non-linear Fitting ("Parent") Functions

Suppose you have a set of data points  $\{x_i, y_i, \sigma_i\}$ . The fitting functions contain fit parameters  $\{a_j\}$ . Some are *linear* in the fit parameters and some are *non-linear* in the parameters.

-  $y_{\text{fit}} = a_0 + a_1 x + a_2 x^2 + a_3 x^3$  ----- Linear

-  $y_{\text{fit}} = a_1 (x^3 - x) + a_2 \sin x + a_3 e^x$  ----- Linear

-  $y_{\text{fit}} = a_1 \sin(a_2 x + a_3) + a_4$  ----- Non-Linear

-  $y_{\text{fit}} = a_1 (x - a_2) + a_3 (x^2 - a_4)^3$  ----- Non-Linear

-  $y_{\text{fit}} = a_0 P_0(x) + a_1 P_1(x) + a_2 P_2(x) + a_3 P_3(x)$  ----- Linear

where the  $P_j$ 's are the Legendre polynomials

**Linear** → solve for  $\{a_i\}$  in one iteration using least-squares fitting à la Bevington Chapter 6

**Non-linear** → solve for  $\{\delta a_i\}$  using least-squares fitting à la Bevington Chapter 8 and iterate as needed

# mpl\_DATAFIT for non-linear fits

Modern Physics Lab Data Fitting

Date Now: \_\_\_\_\_ Time Now: \_\_\_\_\_ Log Book Page: \_\_\_\_\_

R.A.Sch. v.10b

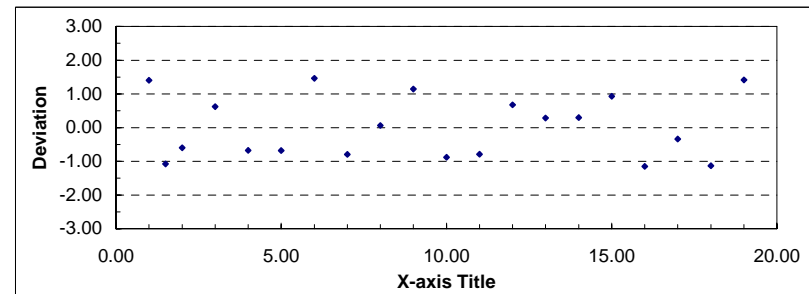
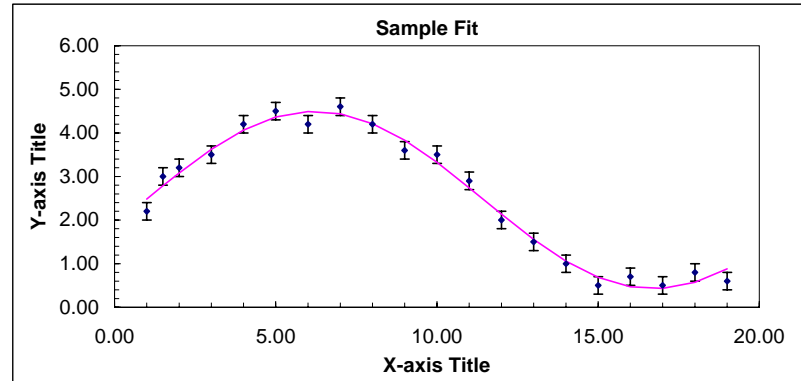
Entries	Constraints	d.o.f.	$\sigma(\chi^2/\text{dof})$
20	4	16	0.35

Substitute your data (up to 1090 lines) below. Program "FIT Theory" column with your fitting function.

	X	Y	DY	FIT	Deviation	Dev. <sup>2</sup>
	Data	Data	Data	Theory		
1	1.00	2.20	0.20	2.48	1.40	1.97
2	1.50	3.00	0.20	2.78	-1.08	1.16
3	2.00	3.20	0.20	3.08	-0.60	0.36
4	3.00	3.50	0.20	3.62	0.62	0.39
5	4.00	4.20	0.20	4.07	-0.67	0.45
6	5.00	4.50	0.20	4.36	-0.68	0.47
7	6.00	4.20	0.20	4.49	1.46	2.13
8	7.00	4.60	0.20	4.44	-0.80	0.64
9	8.00	4.20	0.20	4.21	0.06	0.00
10	9.00	3.60	0.20	3.83	1.15	1.31
11	10.00	3.50	0.20	3.32	-0.88	0.77
12	11.00	2.90	0.20	2.74	-0.79	0.62
13	12.00	2.00	0.20	2.13	0.67	0.46
14	13.00	1.50	0.20	1.56	0.28	0.08
15	14.00	1.00	0.20	1.06	0.30	0.09
16	15.00	0.50	0.20	0.69	0.93	0.86
17	16.00	0.70	0.20	0.47	-1.15	1.32
18	17.00	0.50	0.20	0.43	-0.34	0.12
19	18.00	0.80	0.20	0.57	-1.13	1.28
20	19.00	0.60	0.20	0.88	1.41	2.00

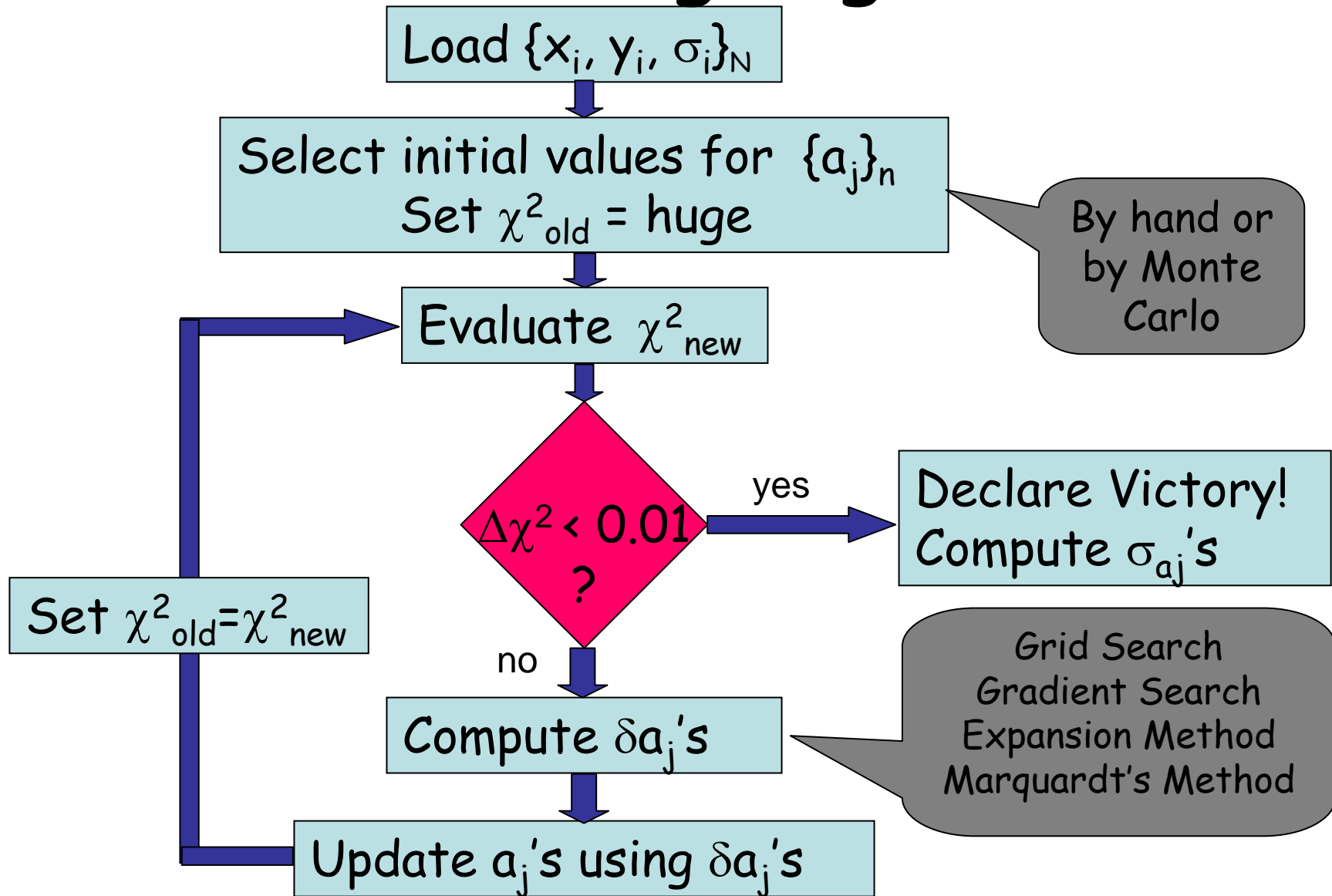
Any function here...

	Value	Initial Step Size	Final Step Size	Estimated Uncertainty	Refined Uncertainty
	Fit Parameter 1=	-2.04	0.1		
Fit Parameter 2=	0.30	0.1			
Fit Parameter 3=	2.85	0.1			
Fit Parameter 4=	2.46	0.1			
Fit Parameter 5=					
$\chi^2 =$	16.50				
$\chi^2/\text{d.o.f.} =$	1.03				

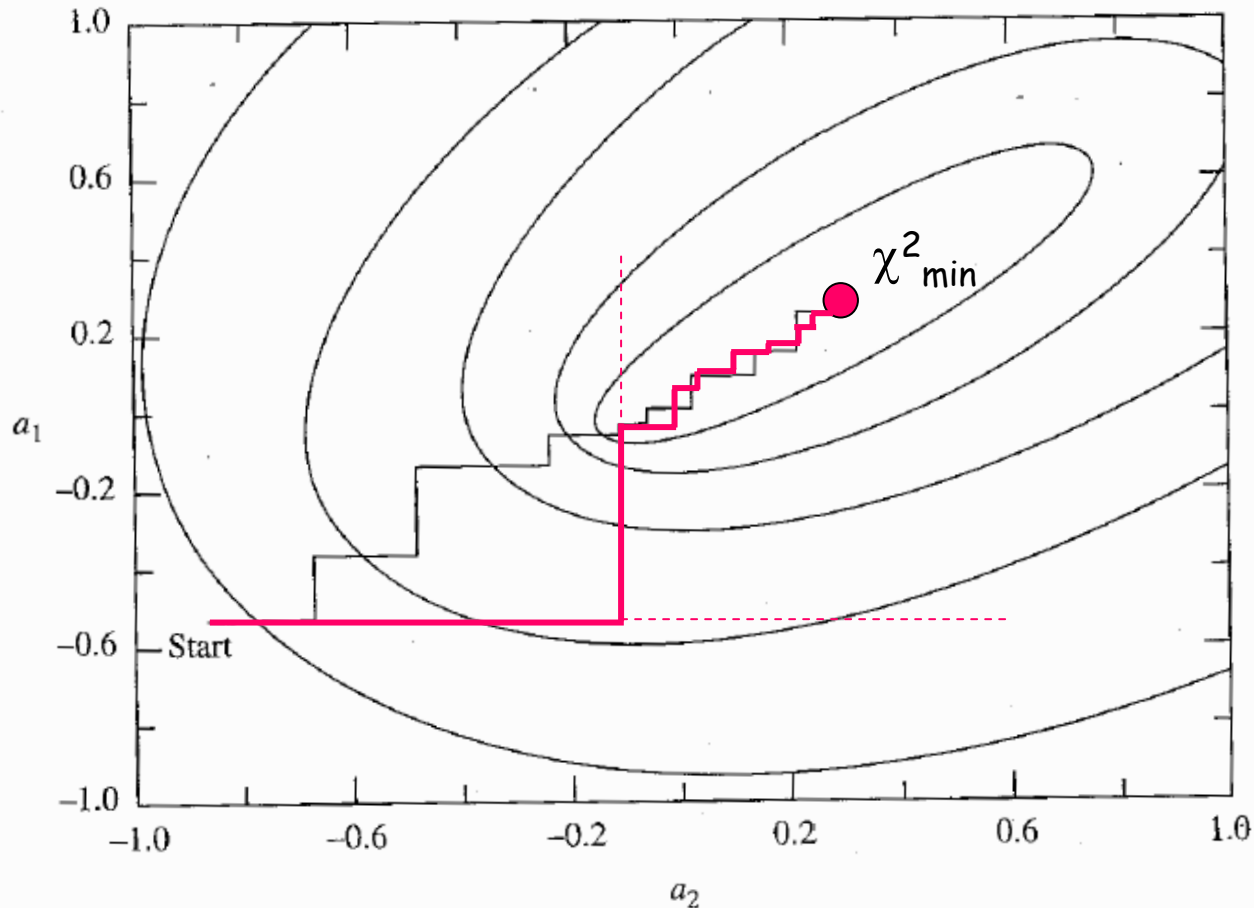


$\chi^2$  Tolerance = +- 0.010

# Generic Fitting Algorithm



# Grid Search (Bevington 8.3; pp.151)



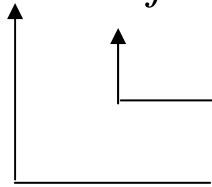
**FIGURE 8.4**

Contour plot of  $\chi^2$  as a function of two highly correlated variables. The zigzag line represents the search path approach to a local minimum by the grid-search method.

# Grid Search (Bevington 8.3; pp.151)

- Treat each  $a_j$  in turn

- Let  $a_j \rightarrow a_j \pm \Delta a_j$



Must pre-select step size

Pick sign such that  $\chi^2$  decreases

- Take repeated steps until  $\chi^2$  increases
- Reverse direction, cut step size in half, etc...
- Pro's and Con's:
  - Simple to program (`mpl_DATAFIT`)
  - Can be slow if parameters are highly correlated
  - User must pick starting step sizes intelligently

# Gradient Search (Bevington 8.4; pp.153)

- Find the "steepest" descent down the valley of  $\chi^2$  by computing the normalized gradient
  - Gradient always computable numerically
  - Sometimes can be computed algebraically

# Gradient Search (Bevington 8.4; pp.153)

$$\nabla\chi^2 = \sum_{j=1}^n \left[ \frac{\partial\chi^2}{\partial a_j} \hat{a}_j \right]$$

$$(\nabla\chi^2)_j = \frac{\partial\chi^2}{\partial a_j} \simeq \frac{\chi^2(a_j + f\Delta a_j) - \chi^2(a_j)}{f\Delta a_j} \quad (8.15)$$

Construct a dimensionless gradient using:  $b_j = \frac{a_j}{\Delta a_j}$

$$\frac{\partial\chi^2}{\partial b_j} \simeq \frac{\chi^2(a_j + f\Delta a_j) - \chi^2(a_j)}{f\Delta a_j} \Delta a_j = \frac{\chi^2(a_j + f\Delta a_j) - \chi^2(a_j)}{f} \quad (8.18)$$

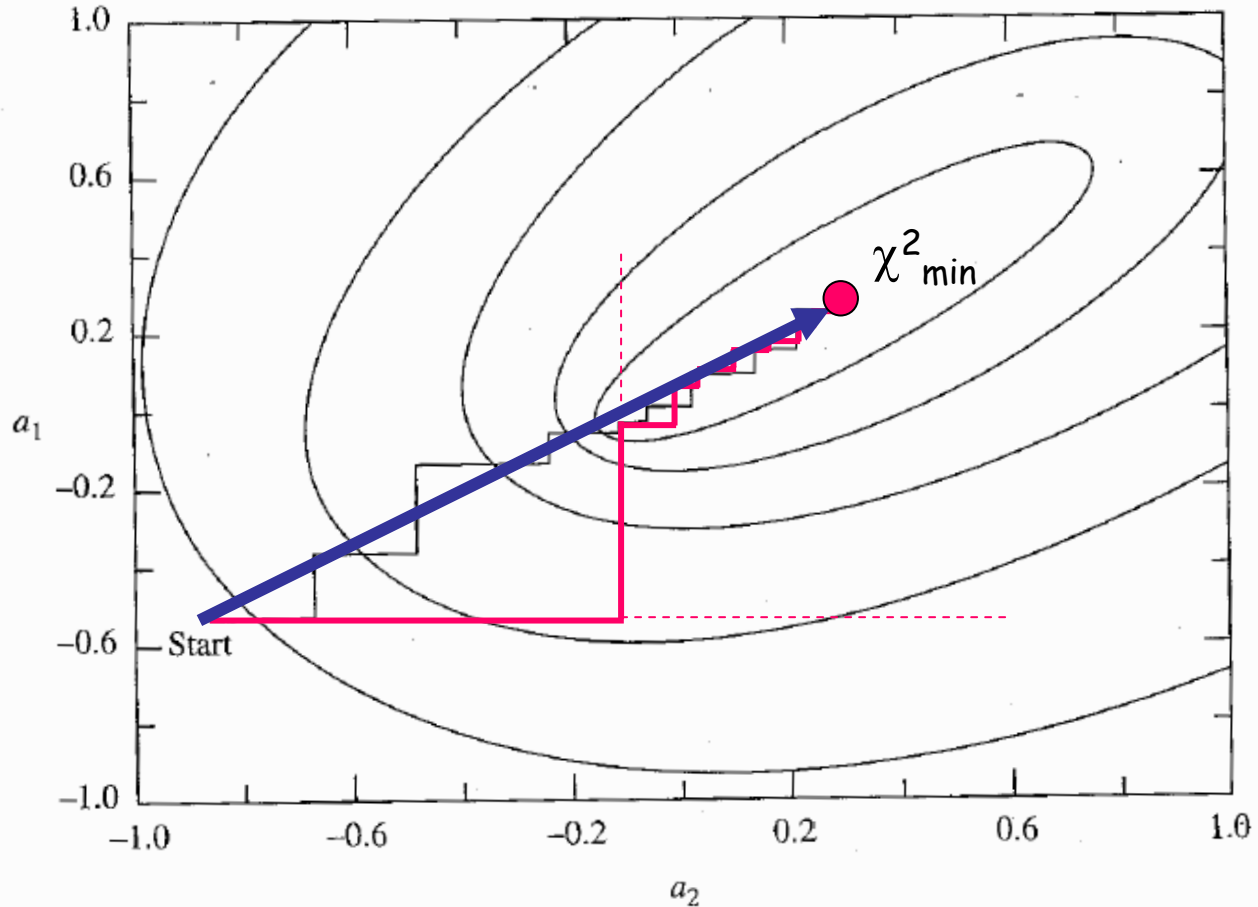
The gradient vector:

$$\gamma_j = \frac{\partial\chi^2/\partial b_j}{\sqrt{\sum_{j=1}^m (\partial\chi^2/\partial b_j)^2}}$$

The parameter increment:

$$\delta a_j = -\gamma_j \Delta a_j$$

# Gradient Method pay-off:



**FIGURE 8.4**

Contour plot of  $\chi^2$  as a function of two highly correlated variables. The zigzag line represents the search path approach to a local minimum by the grid-search method.

# Gradient Search (Bevington 8.4; pp.153)

- Find the “steepest” descent down the valley of  $\chi^2$  by computing the normalized gradient
  - Gradient always computable numerically
  - Sometimes can be computed algebraically
- Pro's and Con's:
  - Fast approach to the minimum from far away
  - Very slow near the minimum where the gradient is near zero

# Summary

- Linear vs. non-linear fitting functions
  - Linear functions can be fitted in one algebraic iteration
  - Non-linear function require iterated solutions when fitting to data
- Grid search is simple but tends to be slow
- Gradient search is fast when far away from minimum but slow near the minimum
- We need a method that is fast near the minimum...next time