UNIVERSITY OF TEXAS ARLINGTON

PHYS 5319-001:

Math Methods in Physics III

Instructor:	**Dr. Qiming Zhang**
Office:	**CPB 336**
Phone:	**817-272-2020**
Email:	**zhang@uta.edu**

Errors and Uncertainties

Supposing a logic flow of a program is: Start→ U_1 → U_2 → ……→ U_n → end

If each unit has probability *p* of being correct, the whole program has the chance $P=p^n$ to be correct. Say we have 1000 units (iterations, steps,…), and *p*=99.93%=0.9993

$P=(0.9993)^{1000} = 0.4965 \approx 50\%$

4 general types of errors

- **Blunders:** typo, graphical error in program data input
- **Random errors:** environments, weather, ……

  solution: fault-tolerance, repeat check
- **Approximation/Algorithm error**

  e.g., from truncation $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \approx \sum_{n=0}^{N} \frac{x^n}{n!} + \varepsilon(x, N)$
- **Roundoff error:** a number stored by machine is limited

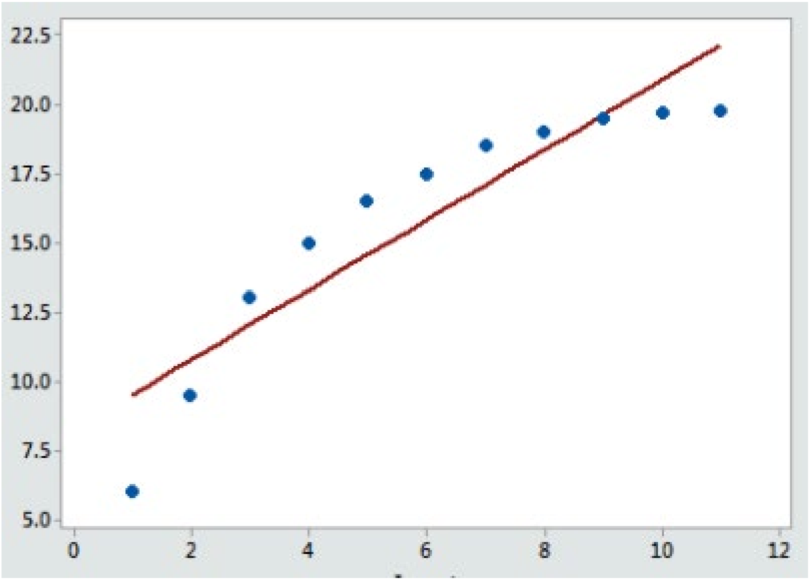
e.g. e^{-x} for very large *x*, e^{-x} is small but x is large, some cancellation may occur

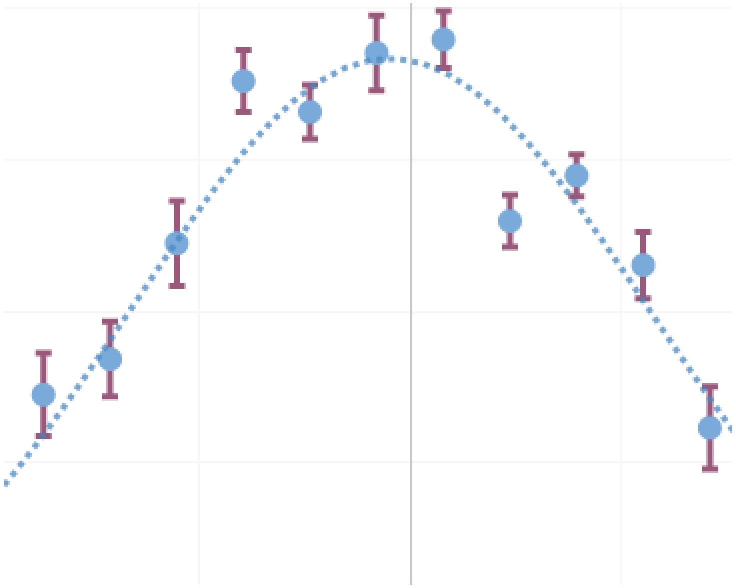
-Subtractive cancelation: If $x_{large} - y_{large} = z_{small}$

the significant figures in z_{small} will be less!

Data Fitting

examples





Data from measurements ($\{x_i, y_i\}$) has deviation from the true values.

How to obtain the law underneath? $y = f(x)$

Lagrange interpolation

In a local/small region, $f(x)$ can be approximated as a polynomial.
In i-th region: $f_i(x) \approx a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$

We have n set of $\{x_i, y_i\}$ $\rightarrow$ (n-1) order polynomial
Formula: $f(x) \approx y_1 \lambda_1(x) + y_2 \lambda_2(x) + \cdots + y_n \lambda_n(x)$

where each $\lambda_i(x) = \prod_{j \neq i}^{n} \frac{x - x_j}{x_i - x_j} = \frac{x - x_1}{x_i - x_1} \frac{x - x_2}{x_i - x_2} \ldots \frac{x - x_n}{x_i - x_n}$

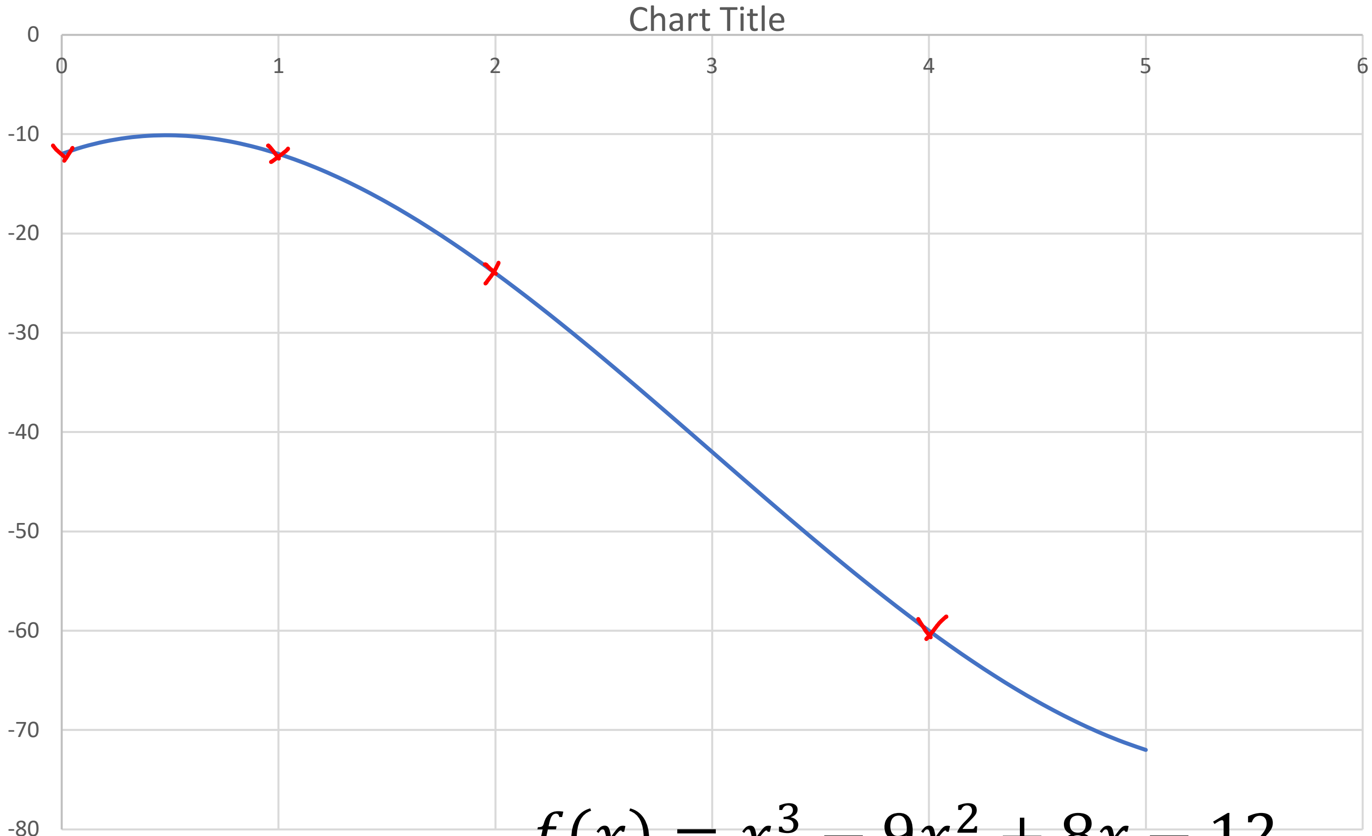
Example: ***i*** **=**	**1**	**2**	**3**	**4**
x_i	**0**	**1**	**2**	**4**
y_i	**-12**	**-12**	**-24**	**-60**

$$f(x) = \frac{(x-1)(x-2)(x-4)}{(0-1)(0-2)(0-4)}(-12) + \frac{(x-0)(x-2)(x-4)}{(1-0)(1-2)(1-4)}(-12) + \cdots$$

$$\Rightarrow \quad f(x) = x^3 - 9x^2 + 8x - 12$$

When $x = x_i$?

Example: i =	**1**	**2**	**3**	**4**
x_i	**0**	**1**	**2**	**4**
y_i	**-12**	**-12**	**-24**	**-60**



$$f(x) = x^3 - 9x^2 + 8x - 12$$

A much better algorithm: Neville's algorithm

"tableau"

$$
\begin{array}{lllll}
x_1: & y_1 = P_1 & & & \\
 & & P_{12} & & \\
x_2: & y_2 = P_2 & & P_{123} & \\
 & & P_{23} & & P_{1234} \\
x_3: & y_3 = P_3 & & P_{234} & \\
 & & P_{34} & & \\
x_4: & y_4 = P_4 & & &
\end{array}
$$

parent

daughter: $P_{12} = y_1 \frac{x - x_2}{x_1 - x_2}$

Recurrence:

$$P_{i(i+1)\dots(i+m)} = \frac{(x - x_{i+m})P_{i(i+1)\dots(i+m-1)} + (x_i - x)P_{(i+1)(i+2)\dots(i+m)}}{x_i - x_{i+m}}$$

for $i = 1, 2, \dots, n-1$

Small difference between parents & daughters

$$C_{m,i} \equiv P_{i\dots(i+m)} - P_{i\dots(i+m-1)}$$

$$D_{m,i} \equiv P_{i\dots(i+m)} - P_{(i+1)\dots(i+m)}$$

$\Rightarrow$

$$D_{m+1,i} = \frac{(x_{i+m+1} - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}}$$

$$C_{m+1,i} = \frac{(x_i - x)(C_{m,i+1} - D_{m,i})}{x_i - x_{i+m+1}}$$

```fortran
      SUBROUTINE POLINT(XA,YA,N,X,Y,DY)
      PARAMETER (NMAX=10)
      DIMENSION XA(N),YA(N),C(NMAX),D(NMAX)
      NS=1
      DIF=ABS(X-XA(1))
      DO I=1,N
        DIFT=ABS(X-XA(I))
        IF (DIFT.LT.DIF) THEN
          NS=I
          DIF=DIFT
        ENDIF
        C(I)=YA(I)
        D(I)=YA(I)
      ENDDO
      Y=YA(NS)
      NS=NS-1
      DO M=1,N-1
        DO I=1,N-M
          HO=XA(I)-X
          HP=XA(I+M)-X
          W=C(I+1)-D(I)
          DEN=HO-HP
          IF(DEN.EQ.0.)PAUSE
          DEN=W/DEN
          D(I)=HP*DEN
          C(I)=HO*DEN
        ENDDO
        IF (2*NS.LT.N-M)THEN
          DY=C(NS+1)
        ELSE
          DY=D(NS)
          NS=NS-1
        ENDIF
        Y=Y+DY
      ENDDO
      RETURN
      END
```

Largest anticipated values of n

If P(x) is the polynomial of degree N-1 such that $P(Xa_i)$-Ya_i, then the returned value Y=P(x)

Initiate C's & D's

Initiate Y

For each column of the tableau, loop over the current C's & D's and update them

updated C's and D's

corrections

Least-square fitting

Looking for the "best" fit

- If errors exist, the "best" fit in a statistical sense won't pass through the data points;
- Only for the simplest case, e.g. linear, we can write down solutions. Usually in library such as SLATEC

netlib.org

Imagine we have N_D values of the independent *y* as a function of the independent *x*:

$$(x_i, y_i \pm \sigma_i), i = 1,2, \ldots, N_D$$

where is the uncertainty (error-bar).

Goal: how well a math function (theory) can describe these data

$f(x)$ contains parameters $\left\{a_1, a_2, \ldots, a_{M_p}\right\}$

Find the best values for $\{a_m\}, i = 1,2, \ldots, M_p$

$$f(x) = f(x; \{a_m\})$$

Set up $\chi^2 \equiv \sum_{i=1}^{N_D} \left(\frac{y_i - f(x_i; \{a_m\})}{\sigma_i} \right)^2$ weight

$$\frac{1}{\sqrt{N}}$$

From expt.: $(\boldsymbol{x_i}, \boldsymbol{y_i} \pm \boldsymbol{\sigma_i}), \boldsymbol{i} = \mathbf{1}, \mathbf{2}, \dots, \boldsymbol{N_D}$
Sum over $\boldsymbol{N_D}$ **points**

Better fit $\Rightarrow$ smaller χ^2

Best fit $\Rightarrow$ minimal χ^2

Note: $\frac{1}{\sigma_i}$ (square) works as a weight here. A less accurate point contributes less.

$$\frac{\partial \chi^2}{\partial a_m} = 0 \; (m = 1, M_p)$$

Obtain M_p equations

i.e.$\Rightarrow$

$$\sum_{i=1}^{N_D} \frac{y_i - f(x_i)}{\sigma_i^2} \frac{\partial f(x_i; \{a_m\})}{\partial a_m} = 0 \quad (m = 1, M_p) \qquad (1)$$

For example: fit $f(x; a_1, a_2) = a_1 + a_2 x$

$\frac{\partial f}{\partial a_1} = 1$

Input: $\{x_i, y_i \pm \sigma_i\}$

a_1 to eq.(1):): $\sum \frac{y_i - a_1 - a_2 x_i}{\sigma_i^2} = 0$

S_x

S_y

$$\left(\sum \frac{y_i}{\sigma_i^2}\right) - \left(\sum \frac{1}{\sigma_i^2}\right) a_1 - \left(\sum \frac{x_i}{\sigma_i^2}\right) a_2 = 0 \qquad (2)$$

S

a_2 to eq.(1): $\sum \frac{y_i - a_1 - a_2 x_i}{\sigma_i^2} x_i = 0$

S_{xx}

S_{xy}

or $$\left(\sum \frac{x_i y_i}{\sigma_i^2}\right) - \left(\sum \frac{x_i}{\sigma_i^2}\right) a_1 - \left(\sum \frac{x_i^2}{\sigma_i^2}\right) a_2 = 0 \qquad (3)$$

S_x

Let S’s to represent $\sum 's$

$$S_y - Sa_1 - S_x a_2 = 0$$
$$S_{xy} - S_x a_1 - S_{xx} a_2 = 0$$

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

$$a_1 = \frac{S_{xx} S_y - S_x S_{xy}}{\Delta}, a_2 = \frac{S S_{xx} - S_x S_y}{\Delta}$$