BPhO Computational Challenge

Seminar 01: Incorporating experimental error in calculations

Dr Andrew French.
December 2021.

BPhO British Physics Olympiad

Standard form

Very small and very large quantities are tedious (and *error prone*) to write out using full decimal notation.

Standard form: e.g. $\boxed{6.67\times 10^{-11}}$

is an *integer* between 1 and 9 followed by $N - 1$ digits, where N is the number of **significant figures** of the quantity.

The power of 10 (the 'exponent') gives you an *immediate sense of scale*.

1 light-year $$1\,\mathrm{ly} = 9.461\times 10^{15}\,\mathrm{m}$$

proton radius $$r \approx 8.42\times 10^{-16}\,\mathrm{m}$$

Precision. A precise measurement is performed to a high number of significant figures. This typically means the *random error* in the measurement (i.e. the ***standard deviation***) is *very small* compared to the ***mean value***. In calculations, one should quote a answer to the *worst precision* (i.e. lowest number of significant figures) of the *input values.*

$$x = 123.4, \quad y = 56.7, \quad z = 8.9$$

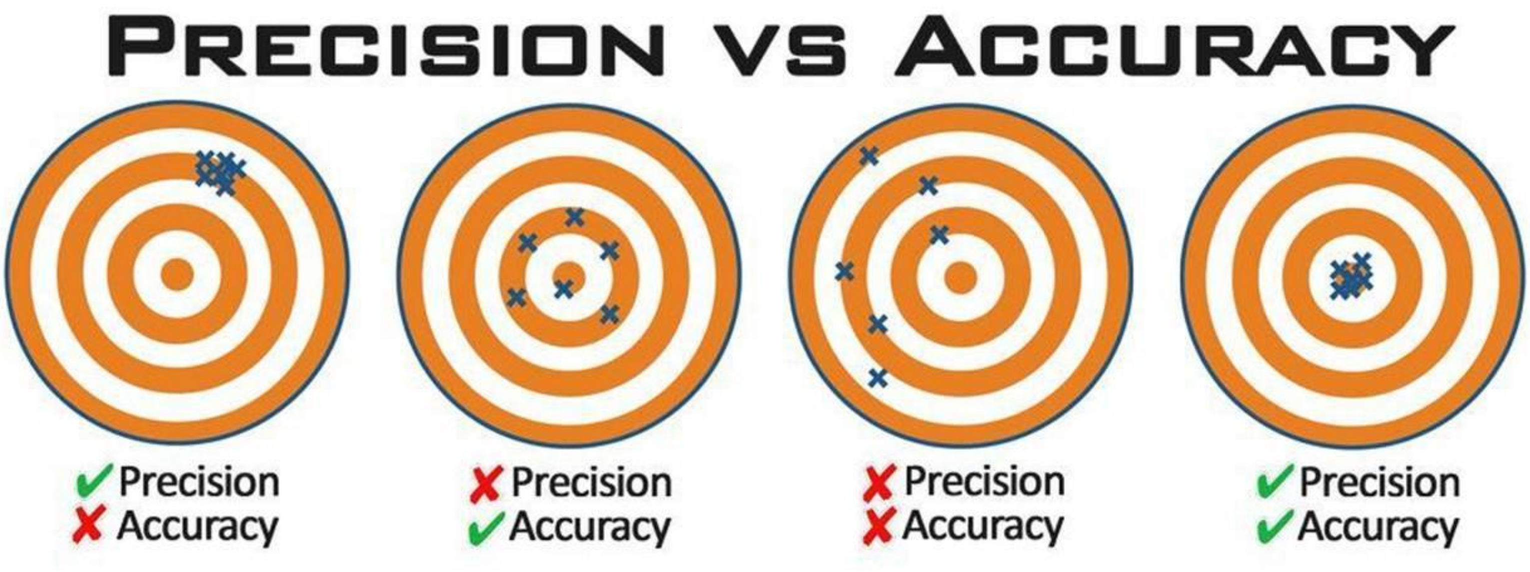
$$\therefore x = 1.234 \times 10^2, \quad y = 5.67 \times 10^1, \quad z = 8.9$$

lowest precision
i.e. 2 s.f.

$$a = \frac{xy}{z} = \frac{123.4 \times 56.7}{8.9} = 786.1550.... \quad \text{(unrounded)}$$

$$\boxed{a = 7.9 \times 10^2 \quad \text{to 2.s.f}}$$

Accuracy relates to the degree of *systematic error*. A time of 12.345s may be *very precise*, but could easily be 2.000s out from a true value of 10.345s if there is some form of accidental offset in the timing system.



Mean and standard deviation

If you have a *sample* of data, which you believe represents a quantity x subject to *random error*:

$$\boxed{\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i}$$

is an *unbiased estimator* of the **mean value** of the quantity x.
N is the number of measurements, and x_i is the i^{th} measurement.

$$\boxed{\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2}}$$

is an *unbiased estimator* of the **error** in this measurement. This is *not quite* the *standard deviation*, which involves an N factor rather than N - 1 in the fraction preceding the sum.

The measurement x can therefore be quoted: $\boxed{x = \bar{x} \pm \sigma_x}$

ERROR CALCULATION

Excel spreadsheet example

ACTUAL X VALUE 123

X VALUES WITH RANDOM ERROR

121	121	125	122	120	128	120	121	124	119

N

10

MEAN X

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

122

$$\left(x_i - \bar{x}\right)^2$$

1.21	1.21	8.41	0.01	4.41	34.8	4.41	1.21	3.61	9.61

ERROR IN X

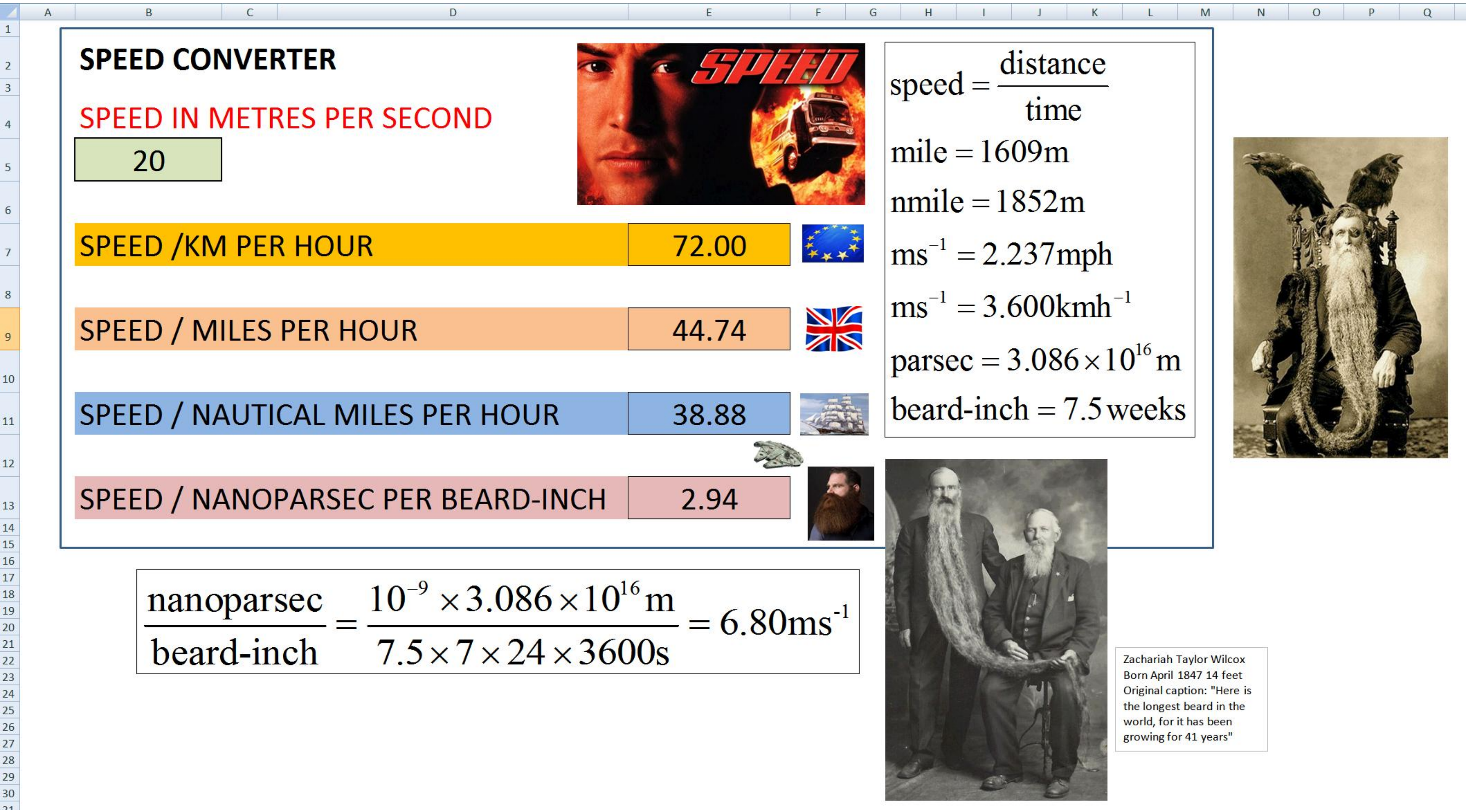
$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2}$$

3

SO: X = (122 +/- 3)

The presentation of numeric information can be as important as the numbers themselves. CLARITY is key. Not only do you want to efficiently compute a number from variable inputs, *you need a sanity check that the calculation is correct.*

Use a spreadsheet to construct a **visual calculator.**



Errors. All measurable quantities will be subject to *uncertainty*. If quantities $x,y....$ are within a known range, we can use **upper and lower bounds** to determine the range of combined quantities.

e.g. $$x_- \le x \le x_+ \qquad y_- \le y \le y_+$$

Therefore: $$\boxed{x_-^2 y_- \le x^2 y \le x_+^2 y_+} \qquad \boxed{x_-^2 / y_+ < x^2 / y < x_+^2 / y_-}$$

Note the mixing of *upper and lower bounds* in the last example.

Another example: $$1.23 \le x \le 4.56, \;\; 7.89 \le y \le 11.2$$

$$z = \frac{\sqrt{y}}{x}$$

$$\frac{\sqrt{7.89}}{4.56} < z < \frac{\sqrt{11.2}}{1.23}$$

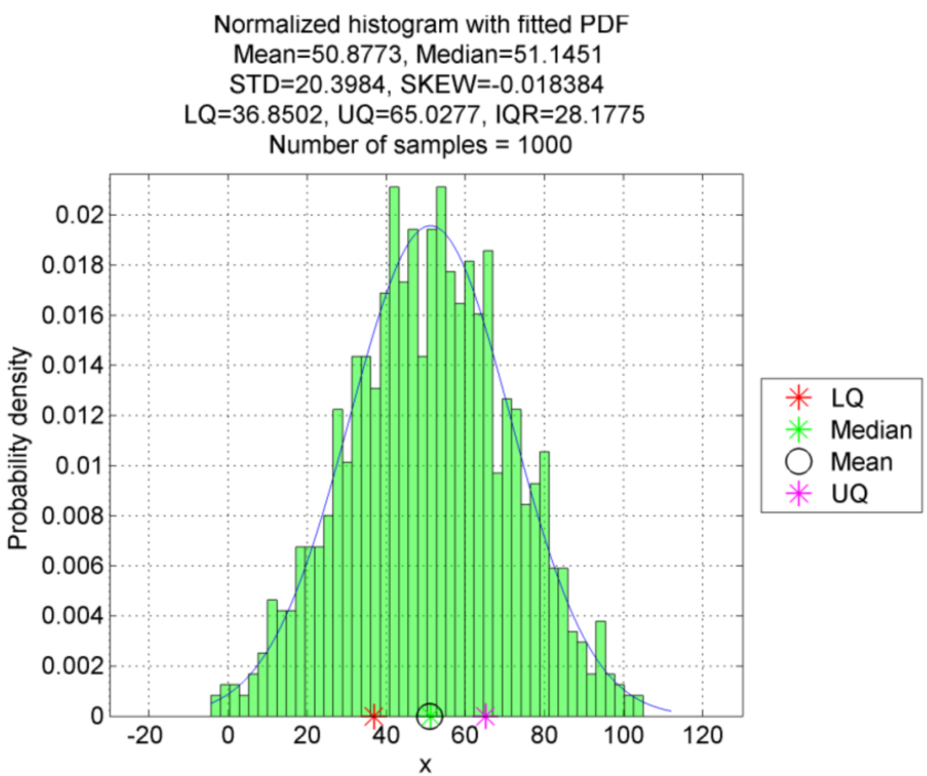
$$\boxed{0.616 < z < 2.721}$$

Laws of Errors – but only if you think errors are *normally distributed*

If errors are *normally distributed*, the **'Law of Errors'** can be useful (although may result in an artificially tighter uncertainty than upper and lower bounds). Let $f(x, y, z..)$ be a function of measureable quantities e.g. $x = \bar{x} \pm \sigma_x$.

$$f = \bar{f} \pm \sigma_f \text{ where } \bar{f} = f(\bar{x}, \bar{y}, \bar{z}...) \quad : \qquad \sigma_f^2 = \left(\frac{\partial f}{\partial x}\sigma_x\right)^2 + \left(\frac{\partial f}{\partial y}\sigma_y\right)^2 + \left(\frac{\partial f}{\partial z}\sigma_z\right)^2 + ...$$

If $f(x, y....) = kx^a y^b ... \Rightarrow \left(\frac{\sigma_f}{f}\right)^2 = \left(\frac{a\sigma_x}{x}\right)^2 + \left(\frac{b\sigma_y}{y}\right)^2 + ...$ You *add* the (power weighted) squares of fractional errors.



Example:

$$f = 3x^2 y^{-\frac{1}{2}}, \quad x = 20 \pm 3, \quad y = 40 \pm 5$$

$$\therefore \left(\frac{\sigma_f}{\bar{f}}\right)^2 = \left(\frac{2\sigma_x}{\bar{x}}\right)^2 + \left(\frac{\frac{1}{2}\sigma_y}{\bar{y}}\right)^2$$

$$\bar{f} = 3 \times 20^2 \times 40^{-\frac{1}{2}} = 189.7366....$$

$$\therefore \sigma_f = \bar{f}\sqrt{\left(\frac{2 \times 3}{20}\right)^2 + \left(\frac{\frac{1}{2}5}{40}\right)^2} = 58.14....$$

$$\therefore f = (1.9 \pm 0.6) \times 10^2$$

UPPER AND LOWER BOUND CALCULATION VS LAWS OF ERRORS

$$f(w,x,y,z) = \frac{kw^a x^b}{y^c z^d}$$

k =	3.00					
w =	1.00	+/-	0.00		**a =**	0.00
x =	20.00	+/-	3.00		**b =**	2.00
y =	40.00	+/-	5.00		**c =**	0.50
z =	1.00	+/-	0.00		**d =**	0.00

Paste as values to here. THEN SAVE BEFORE RUNNING MATLAB CODE.

$$f = 3x^2 y^{-\frac{1}{2}}$$

$$x = 20 \pm 3$$

$$y = 40 \pm 5$$

mean f	fmean =	189.74		
upper bound for f	fmax =	268.25	fmean +	78.52
lower bound for f	fmin =	129.24	fzmean -	60.49
Gaussian error in f		58.14		

$$f(w,x,y,z) = \frac{kw^a x^b}{y^c z^d}$$

$$\sigma_f^2 = \left(\frac{\partial f}{\partial x}\sigma_x\right)^2 + \left(\frac{\partial f}{\partial y}\sigma_y\right)^2 + \left(\frac{\partial f}{\partial z}\sigma_z\right)^2 + \ldots$$

$$f(x,y\ldots) = kx^a y^b \ldots \quad \Rightarrow \left(\frac{\sigma_f}{f}\right)^2 = \left(\frac{a\sigma_x}{x}\right)^2 + \left(\frac{b\sigma_y}{y}\right)^2 + \ldots$$

UPPER AND LOWER BOUND CALCULATION VS LAWS OF ERRORS

$$f(w,x,y,z)=\frac{kw^a x^b}{y^c z^d}$$

k = 3.00

w =	1.00	+/-	0.00	**a =**	0.00
x =	20.00	+/-	3.00	**b =**	2.00
y =	40.00	+/-	5.00	**c =**	0.50
z =	1.00	+/-	0.00	**d =**	0.00

Paste as values to here. THEN SAVE BEFORE RUNNING MATLAB CODE.

mean f	fmean =	189.74		
upper bound for f	fmax =	268.25	fmean +	78.52
lower bound for f	fmin =	129.24	fzmean -	60.49
Gaussian error in f		58.14		

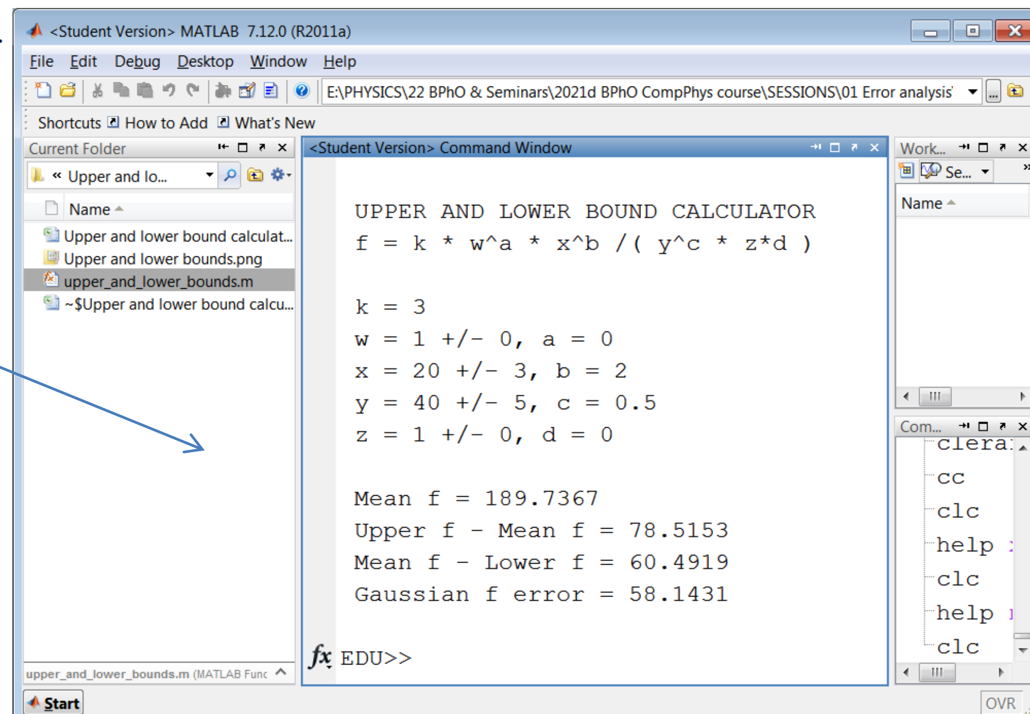
Excel display

$$f(w,x,y,z)=\frac{kw^a x^b}{y^c z^d}$$

$$\sigma_f^2=\left(\frac{\partial f}{\partial x}\sigma_x\right)^2+\left(\frac{\partial f}{\partial y}\sigma_y\right)^2+\left(\frac{\partial f}{\partial z}\sigma_z\right)^2+\ldots$$

$$f(x,y....)=kx^a y^b\ldots \Rightarrow \left(\frac{\sigma_f}{f}\right)^2=\left(\frac{a\sigma_x}{x}\right)^2+\left(\frac{b\sigma_y}{y}\right)^2+\ldots$$

MATLAB command window display



$$z=3x^2y^{-\frac{1}{2}}$$

$$x=20\pm 3$$

$$y=40\pm 5$$

To avoid tedium, we ingest these from the Excel sheet

MATLAB code to evaluate upper and lower bound calculations.

Although you have to ‘think in arrays’, dealing with word-variables is easier than cell references!

```
%Upper and lower bound error calculator
% f = k * w^a * x^b /( y^c * z*d )
function upper_and_lower_bounds
clc; disp(' '); disp(' UPPER AND LOWER BOUND CALCULATOR ')
disp(' f = k * w^a * x^b /( y^c * z*d ) '); disp(' ')

%Get parameters from Excel
[k,w,x,y,z,sigma_w,sigma_x,sigma_y,sigma_z,a,b,c,d] = get_data_from_excel;

%Calculate outputs
mean_f = k * (w^a) * (x^b) /( (y^c) * (z^d) );
f_upper = k *...
    ( ( (w+sigma_w)^a ) * ( (x+sigma_x)^b ) ) /...
    ( ( ( y - sigma_y)^c ) * (( z-sigma_z)^d ) );
f_lower = k *...
    ( ( (w-sigma_w)^a ) * ( (x-sigma_x)^b ) ) /...
    ( ( ( y + sigma_y)^c ) * (( z+sigma_z)^d ) );

gaussian_f_error = mean_f *...
    sqrt( ( a*sigma_w/w)^2 + ( b*sigma_x/x)^2 +...
    ( c*sigma_y/y)^2 + ( d*sigma_z/z)^2 );

%Display message to command window
disp([' k = ',num2str(k) ]);
disp([' w = ',num2str(w),' +/- ',num2str(sigma_w),', a = ',num2str(a)]);
disp([' x = ',num2str(x),' +/- ',num2str(sigma_x),', b = ',num2str(b) ]);
disp([' y = ',num2str(y),' +/- ',num2str(sigma_y),', c = ',num2str(c) ]);
disp([' z = ',num2str(z),' +/- ',num2str(sigma_z),', d = ',num2str(d) ]);
disp(' ');
disp([' Mean f = ',num2str(mean_f) ] );
disp([' Upper f - Mean f = ',num2str(f_upper - mean_f) ] );
disp([' Mean f - Lower f = ',num2str(mean_f - f_lower) ] );
disp([' Gaussian f error = ',num2str(gaussian_f_error) ] ); disp(' ');

%%
```

UPPER AND LOWER BOUND CALCULATION VS LAWS OF ERRORS

k =	3.95				
w =	23.08	+/-	0.55	a =	2.47
x =	60.43	+/-	0.12	b =	2.85
y =	65.94	+/-	0.18	c =	3.36
z =	22.71	+/-	0.69	d =	1.82

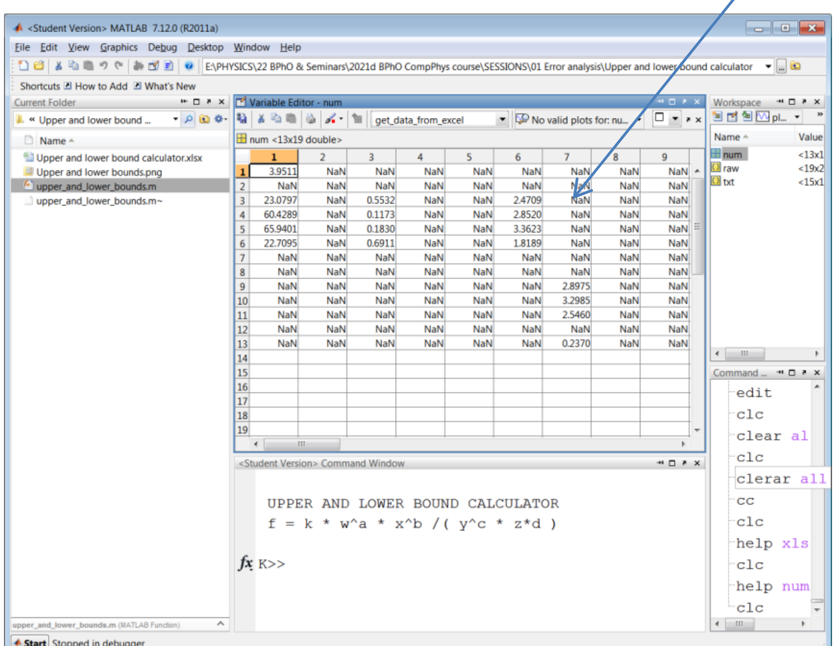
Paste as values to here. THEN SAVE BEFORE RUNNING MATLAB CODE.

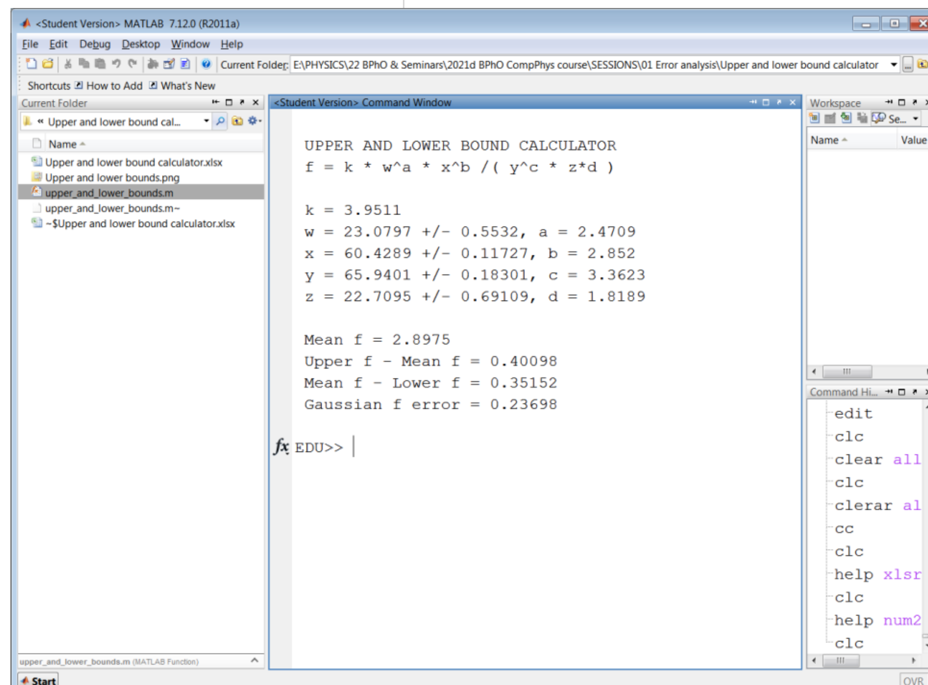
In this example, to avoid tedious hard coding, we ingest the data directly from Excel into MATLAB.

Note we ingest the full precision of the stored numbers. They have been chosen to be displayed to 2.d.p. in Excel.

```
%Get parameters from Excel
function [k,w,x,y,z,sigma_w,sigma_x,sigma_y,sigma_z,a,b,c,d] = get_data_from_excel
[num,txt,raw] = xlsread('upper and lower bound calculator.xlsx');
k = num(1,1);
w = num(3,1); sigma_w = num(3,3); a = num(3,6);
x = num(4,1); sigma_x  = num(4,3); b = num(4,6);
y = num(5,1); sigma_y  = num(5,3); c = num(5,6);
z = num(6,1); sigma_z = num(6,3); d = num(6,6);
```

This function performs the ingest into MATLAB





BPhO Computational Challenge

- Suggested homework
- Q&A

BPhO British Physics Olympiad