Measurement Uncertainty – Part 1 of 2

Why analyze measurement uncertainty?

Types of Measurement Uncertainty

Blunders:

Systematic (or fixed) uncertainty:

Random uncertainty:
Measurement Uncertainty – Part 1 of 2

Estimation of Uncertainty

Single Sample Experiments (aka Type B uncertainty)

Random errors occur due to unknown factors. These errors are not correctable, in general. Blunders and systematic errors can be avoided or corrected. It is the random errors that we must account for in uncertainty analyses. How we quantify random errors depends on whether we conduct a single experiment or multiple experiments. Each case is examined in the following sections.

1. Single Sample Experiments

A single sample experiment is one in which a measurement is made only once. This approach is common when the cost or duration of an experiment makes it prohibitive to perform multiple experiments.

The measure of uncertainty in a single sample experiment is $\pm \frac{1}{2}$ the smallest scale division (or least count) of the measurement device. For example, given a thermometer where the smallest discernable scale division is $1^\circ C$, the uncertainty in a temperature measurement will be $\pm 0.5^\circ C$. If your eyesight is poor and you can only see $5^\circ C$ divisions, then the uncertainty will be $\pm 2.5^\circ C$. One should use an uncertainty within which they are 95% certain that the result lies.

Example: The least count for the ruler to the left is 1 mm. Hence, the uncertainty in the length measurement will be $\pm 0.5$ mm.

Example: You use a manual electronic stop watch to measure the speed of a person running the 100 m dash. The stop watch gives the elapsed time to $1/1000$th of a second. What is the least count for the measurement?

SOLUTION: Although the stop watch has a precision of $1/1000$th of a second, you cannot respond quickly enough to make this the limiting uncertainty. Most people have a reaction time of $1/10$th of a second. (Test yourself by having a friend drop a ruler between your fingers. You can determine your reaction time by where you catch the ruler.) Hence, to be 95% certain of your time measurement, you should use an uncertainty of $\pm \frac{1}{2}(0.1$ sec$) = \pm 0.05$ sec.

Be Sure To:

1. Always indicate the uncertainty of any experimental measurement.
2. Carefully design your experiments to minimize sources of error.
3. Carefully evaluate your least count. The least count is not always $\pm \frac{1}{2}$ of the smallest scale division.

What is the least count for this ruler?

What is the measurement uncertainty using this ruler?

What is the length of the yellow box?

What is the least count for this stopwatch?

What is the measurement uncertainty using this stopwatch?
Measurement Uncertainty – Part 1 of 2

Multiple Sample Experiments (aka Type A uncertainty)

Sample (Arithmetic) Mean (\(\bar{x}\)):

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

Sample Variance (\(s^2\)) and Sample Standard Deviation (\(s\)):

\[ s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2 \]

\[ s \approx \sqrt{s^2} \]

The Normal (aka Gaussian) Probability Distribution:

\[ p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left( -\frac{(x-\mu)^2}{2\sigma^2} \right) \]

\[ \int_{-\infty}^{\infty} p(x) dx = 1 \]

<table>
<thead>
<tr>
<th>#</th>
<th>#</th>
<th>#</th>
<th>#</th>
<th>#</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_i</td>
<td>x_i</td>
<td>x_i</td>
<td>x_i</td>
<td>x_i</td>
<td>x_i</td>
</tr>
<tr>
<td>1</td>
<td>99.36</td>
<td>120.20</td>
<td>80.92</td>
<td>72.20</td>
<td>130.41</td>
</tr>
<tr>
<td>2</td>
<td>121.02</td>
<td>76.56</td>
<td>88.64</td>
<td>95.92</td>
<td>116.38</td>
</tr>
<tr>
<td>3</td>
<td>131.73</td>
<td>93.18</td>
<td>93.32</td>
<td>100.04</td>
<td>113.11</td>
</tr>
<tr>
<td>4</td>
<td>119.56</td>
<td>65.21</td>
<td>98.33</td>
<td>82.72</td>
<td>103.82</td>
</tr>
<tr>
<td>5</td>
<td>94.31</td>
<td>105.08</td>
<td>143.42</td>
<td>72.67</td>
<td>102.18</td>
</tr>
<tr>
<td>6</td>
<td>114.74</td>
<td>102.21</td>
<td>116.85</td>
<td>147.12</td>
<td>101.71</td>
</tr>
<tr>
<td>7</td>
<td>78.33</td>
<td>76.47</td>
<td>98.88</td>
<td>104.78</td>
<td>96.41</td>
</tr>
<tr>
<td>(\bar{x})</td>
<td>108.44</td>
<td>91.27</td>
<td>102.91</td>
<td>96.49</td>
<td>109.15</td>
</tr>
<tr>
<td>(s^2)</td>
<td>342.05</td>
<td>377.60</td>
<td>442.04</td>
<td>665.40</td>
<td>135.73</td>
</tr>
<tr>
<td>(s)</td>
<td>18.49</td>
<td>19.43</td>
<td>21.02</td>
<td>25.80</td>
<td>11.65</td>
</tr>
</tbody>
</table>

\[ \mu = 100 \quad \sigma^2 = 400 \quad (\sigma = 20) \]

\[ \# \quad x_i \]

- Frequency plot of the trial mean values (\(\bar{x}\)).
- Superimposed normal distribution using:
  \[ \mu = \text{mean of the trial means} \]
  \[ \sigma = s_{\bar{x}} = 6.99 \]
**What's the difference between the sample standard deviation and the standard error of the sample mean?**

**What happens as the number of measurements increases?**

**What happens as the sample standard deviation decreases?**

**What’s the difference between the sample standard deviation and the standard error of the sample mean?**
How to find the standard error in the fitting parameters?

In Excel: Use the LINEST command, e.g., LINEST(B2:B15, A2:A15, TRUE, TRUE)

Excel output:

<table>
<thead>
<tr>
<th></th>
<th>Fit for m</th>
<th>Fit for b</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.98241758</td>
<td>3.54285714</td>
<td></td>
</tr>
<tr>
<td>0.08845073</td>
<td>0.676518</td>
<td></td>
</tr>
<tr>
<td>0.9962324</td>
<td>1.33411149</td>
<td></td>
</tr>
<tr>
<td>3173.05351</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>5647.57033</td>
<td>21.3582418</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fit for m</th>
<th>Fit for b</th>
</tr>
</thead>
<tbody>
<tr>
<td>std error for m</td>
<td>std error for b</td>
</tr>
<tr>
<td>coef. of det., R²</td>
<td>std. error for y, y</td>
</tr>
<tr>
<td>regression sum of squares</td>
<td>residual sum of squares</td>
</tr>
</tbody>
</table>

In Python:

```python
# Least squares fit, including standard error estimates.
import numpy as np
import scipy.optimize as opt
import matplotlib.pyplot as plt

# The data to fit.
x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13])
y = np.array([2, 9, 14, 16, 25, 29, 34, 40, 44, 48, 53, 59, 64, 66])

# The equation to fit the data to.
def func(x, m, b):
    return m*x + b

# Guess initial estimates for the parameters to seed the curve fit.
p0 = [1, 1]  # [m, b] These are just reasonable guesses.
popt, pcov = opt.curve_fit(func, x, y, p0)

# Print the fitting parameter values.
p = np.sqrt(np.diag(pcov))
print("m = %.3e\n\nb = %.3e\n\nstandard error for m = %.3e\n\nstandard error for b = %.3e\n" % tuple(popt))
```

Python output:

```
m = 4.982e+00
b = 3.543e+00
standard error for m = 8.845e-02
standard error for b = 6.765e-01
```

What is the uncertainty on the (least squares) curve fit slope (m) and intercept (b)?

Answer:

\[ m \pm 1.96s_m, \text{ where } s_m \text{ is the standard error for } m \]
\[ b \pm 1.96s_b, \text{ where } s_b \text{ is the standard error for } b \]
Measurement Uncertainty – Part 1 of 2

Notes:
1. What is the difference between histograms, frequency distributions, and probability distributions?

**Histograms**
- Plots the number of samples within a specified size bin.
- Plot changes depending on the bin size.
- Area under the curve isn’t equal to one.

**Frequency distribution**
- The fraction of samples in the range \((x, x + \Delta x)\) is the area under the curve in that range, i.e.,
  \[
  \text{fraction of samples in the range } (x, x + \Delta x) = \int_x^{x+\Delta x} f(x) \, dx
  \]
- Plot is insensitive to the bin size.
- Area under the curve is equal to one, i.e.,
  \[
  \int_{-\infty}^{+\infty} f(x) \, dx = 1
  \]
- \(f(x_i, x_i + \Delta x_i) = \frac{1}{\Delta x_i} \frac{n(x_i, x_i + \Delta x_i)}{N}\)

**Probability distribution**
- A frequency distribution, but with an infinite number of samples.
2. How do you know if your data is normally distributed?
   a. Qualitative comparison of frequency distribution to a normal probability distribution with the same mean and standard deviation [not recommended]

   ![Histogram](histogram)
   ![Normal Q-Q plot](normal_q-q_plot)

   Does the data follow a normal distribution?

   Where does the data deviate most from a normal distribution?

   b. Normal Q-Q plot (Normal Quantile-Quantile plot) [qualitative – provides a visual check]

   Quantiles (aka percentiles) are the data values below which a certain proportion of the data fall:
   - 0.1% of the data have values less than $\mu - 3\sigma$
   - 2.2% of the data have values less than $\mu - 2\sigma$
   - 15.8% of the data have values less than $\mu - 1\sigma$
   - 50% of the data have values less than $\mu$
   - 84.2% of the data have values less than $\mu + 1\sigma$
   - 97.8% of the data have values less than $\mu + 2\sigma$
   - 99.9% of the data have values less than $\mu + 3\sigma$

   ![Normal Q-Q plot](normal_q-q_plot)

   Does the data follow a normal distribution?

   Where does the data deviate most from a normal distribution?

   c. Anderson-Darling Goodness-of-Fit Test (quantitative – recommended)

   - Details not presented here – out of scope for this course.
   - Tests if the sample comes from a specified probability distribution, e.g., a normal distribution.
   - Numerically compares the sample distribution to the specified probability distribution.
   - Requires a more advanced knowledge of statistics to apply and understand.
   - There are other, similar quantitative tests of normality, e.g., the Shapiro-Wilks test, the Kolmogorov-Smirnov test, and the Skewness-Kurtosis test.
Example python code to make these various plots and perform the Anderson-Darling test.

```python
# Tests for data normality.
# Disclaimer: I barely know how to program in python. You can probably do better.
import statsmodels.api as sm
import scipy.stats as stats
import numpy as np
import matplotlib.pyplot as plt

# Load the data set from a file (single data entry per line)
my_data = np.loadtxt("data.txt")

# Report some statistics about the data
mean = np.mean(my_data)
stddev = np.std(my_data)
print("# of data entries=", len(my_data))
print("mean = ", mean)
print("std. dev = ", stddev)

# First plot a histogram of the data.
Nbins = 8
plt.figure(1)
plt.hist(my_data, density=False, bins=Nbins, linewidth=1, edgecolor='black')
plt.ylabel('Count')
plt.xlabel('Value')
plt.figure(2)
plt.plot(bin_centers, counts, color='black', marker='o', linestyle='solid')
plt.ylabel('Frequency [1/Value]')
plt.xlabel('Value')

# Include a plot of a normal distribution on top of the frequency distribution.
x = np.linspace(mean-3*stddev, mean+3*stddev,100)
plt.plot(x, stats.norm.pdf(x, mean, stddev), color='red', linestyle='solid')

# Create a QQ plot.
sm.qqplot(my_data, line='s')

# Check data for normality using the Anderson-Darling test.
statistic, significance_values, critical_values = stats.anderson(my_data,'norm')
print("statistic = ", statistic)
print("critical_values = ", critical_values)
print("significance_values = ", significance_values)
for i in range(len(critical_values)):
    if significance_values[i] < 0.05:
        print("The data is NOT consistent with a normal distribution for the critical value of ", critical_values[i])
    else:
        print("The data IS consistent with a normal distribution for the critical value of ", critical_values[i])
plt.show()
```

Python output (plots shown previously):

```
# of data entries= 100
mean = 99.98454800000003
std. dev = 19.693678789517616
statistic = 0.36885781711968946
significance_values = [0.36885781711968946]
critical_values = [15.  10.   5.   2.5  1. ]
```

References

Taylor, J., An Introduction to Error Analysis, University Science Books.