Dice-Measure

Oleg Ryzhov

May 2022

1 Introduction

1.1 Goal of this practice

The goal of this practice task is to work out LaTeX skills and skills in calculating errors for multi-variable function using Python. Of course, for calculation errors we should revise, what are **certain** and **uncertain measurements**, so, let‘s move to the **Theory part**

1.2 Theory part

To start with, let us firstly say, that we measure some physical value, using some *instrument*: we can count high-energy particles in X-ray telescope, we can measure mass with scales and so on. These procedures are called **certain measurements**, when you directly get physical quantity.

Unfortunately, not everything we can get using some instrument, for example, a gravitational force, a distance to galaxies, the temperature on the surface of a star etc. In fact, for these immeasurable quantities we have special formulae, that contain other measurable quantities, so, the strategy is to find these quantities, then take them to the expression, count all that and get your result - these action we call **uncertain measurements**.

Now you should stop and ask two questions. Firstly, usually we have a list of different measures of one quantity, something like $[20.1\,°C, 20.4\,°C, 20.7\,°C, 21.1\,°C]$, so what value from these list should we take to a final formula? Secondly, what should we do with errors of these values, how using them we can get the error of our immeasurable quantity?

Certain measurements

The answer for the first question is called **arithmetical mean**:

$$\overline{x} = \sum_{i=1}^{n} x_i \tag{1}$$

Python-command. You can use library ‘statistics‘ and command ‘.mean(your list of values)‘ to get mean

It‘s our start point for counting any errors, for all kinds of certain and uncertain measurements.

Firstly, let‘s look at the situation with list of certainly (in some cases ’directly’) measured values of some quantity. Second important expression is **standard error (SE)**:

$$S_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}} \tag{2}$$

Python-command. You can use library ‘statistics‘ and command ‘.stdev(your list of values)‘ to get SE

The part under root is called **variance** and usually in literature is signed like σ^2. This values characterise the tendency, how close or far are our numerous values from their mean. And **standard mean error (SEM)** characterises the width of area, where all the values tend to ’appear’:

$$S_{\overline{x}} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n(n-1)}} = \frac{S_x}{\sqrt{n}} \tag{3}$$

Was that step the last in our error-counting for one quantity? Honestly... No, we should add to the SEM some more details, let‘s call them 'additional errors'. So, finally, we got the **'full-SEM'**:

$$U_k(x) = \sqrt{S_{\overline{x}}{}^2 + \frac{1}{3} \cdot (\Delta_d x)^2 + \frac{1}{3} \cdot (\Delta_e x)^2 + \frac{1}{3} \cdot (\Delta_t x)^2}, \tag{4}$$

Where $\Delta_d x$ is error of the instrument, $\Delta_e x$ is error of experimenter, $\Delta_t x$ is error known from literature for these experiment.

Last, if it‘s required, you can count **weights and weighted mean** for your list, but it‘s not asked in our task. So, as answer you get:

$$\overline{x} \pm U_k(x) \tag{5}$$

Uncertain measurements

After we looked through the theory of counting errors for measurable quantities, let's now apply this for answering the second question from introduction. To begin with, it's important to say, that we have two types of multi-variable functions:

- Variables are independent, but have a link between each other
- *Variables are totally independent, even from each other*

Frankly speaking, there are not so many examples of first case, eventually, almost every formula, including our task, contains fully independent variables. That's why we will immediately work out, how to estimate errors for latter, and won't stop at the former.

Generally, our immeasurable quantity is the multi-variable function:

$$\text{f(a}_1, a_2, a_3, a_4, ..., a_n)$$

Of course, we can write all the formulae for numerous variables 'a', but we would rather write expressions for 2 variables (as usual, x and y) to perceive the idea.

Firstly, we should use all the formulae, that we have written earlier (1-4) for x and y, and get $U_k(x)$ and $U_k(y)$. So, what's next?

We will miss all the proof for the expressions lower, but if it's really required to show them, we will do in the next volume

So, to get **full-SEM** for such a function, we should use this expression:

$$U_k(f(x,y)) = \sqrt{(\frac{\partial f}{\partial x})^2 U_k(x)^2 + (\frac{\partial f}{\partial x})^2 U_k(x)^2} \tag{6}$$

Well, in general, for n variables we get **full-SEM**:

$$U_k(f(a_1, a_2, a_3, a_4, ..., a_n)) = \sqrt{\sum_{i=1}^{n} (\frac{\partial f}{\partial a_i})^2 U_k(a_i)^2} \tag{7}$$

2 Dice-theory

Conditions and Measurements

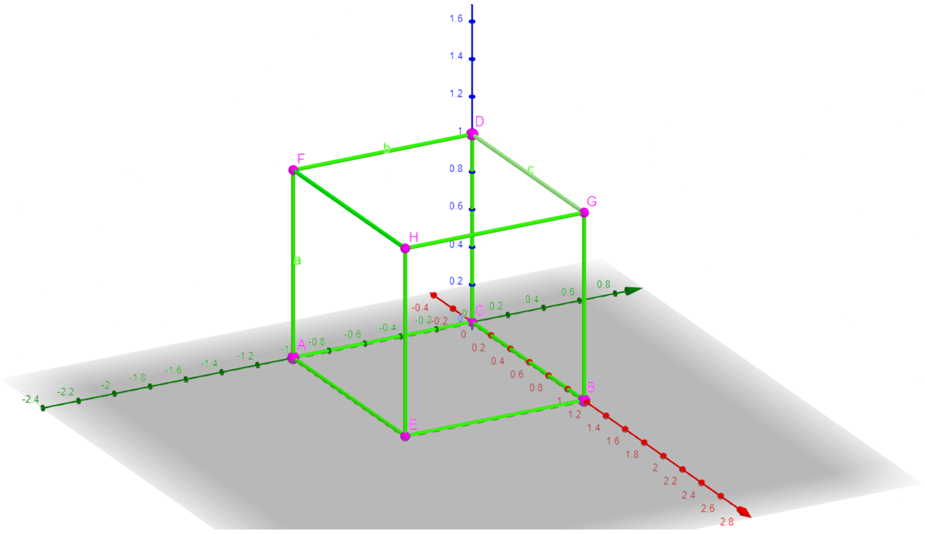

Fig. 1. Our dice, not in scale

Our task was to find out the density ρ of the dice, that you can see at the image upper. We were given the lists of measurements for 3 sides of dice (*a-side, b-side, c-side*) and it's *mass*:

number	a-side, mm	b-side, mm	c-side, mm	mass, g
1	25.12	24.44	15.68	11.4390
2	25.00	24.42	15.70	11.4936
3	25.50	24.60	15.80	11.4397

Of course, it's essentially mark additional errors, at our condition they are:

- $\Delta_d x = 0.02mm$, (here 'x' means sides in mm)
- $\Delta_d m = 0.0001g$
- $\Delta_e m = 0.0004g$

Due to conditions, our theoretically expected result is $\rho_0 = 0.001180\frac{g}{mm^3}$

Required theory

Obviosly, before we will start estimating our errors for density ρ, we should introduce some formulae to take it, for instance, to take partial-derivatives of it for (7). We assume, that our dice is the ideal parallelepipedon, so:

$$\rho = \frac{m}{a \cdot b \cdot c} \tag{8}$$

$$\overline{\rho} = \frac{\overline{m}}{\overline{a} \cdot \overline{b} \cdot \overline{c}} \tag{9}$$

To get means of all our measurable quantities, we simply use (1), then, step by step, from the formula (1) to the last (7) we get **full-SEM** for density ρ. We won't stop at it, and immediately will go to our results, that we have got using Python, only one thing we will mention:

$$\frac{\partial \rho}{\partial m} = \frac{1}{\overline{a} \cdot \overline{b} \cdot \overline{c}} \quad \frac{\partial \rho}{\partial a} = -\frac{\overline{m}}{\overline{a}^2 \cdot \overline{b} \cdot \overline{c}} \quad \frac{\partial \rho}{\partial b} = -\frac{\overline{m}}{\overline{b}^2 \cdot \overline{a} \cdot \overline{c}} \quad \frac{\partial \rho}{\partial c} = -\frac{\overline{m}}{\overline{c}^2 \cdot \overline{b} \cdot \overline{a}} \tag{10}$$

3 Results

Tabular with results

Measure type	**Symbol & units of a quantity**	**Mean, $\overline{x}$**	**SEM, $S_{\overline{x}}$**	**full-SEM, $U_k(x)$**
Certain measurements	a, mm	25.207	0.151	0.151
	b, mm	24.487	0.057	0.058
	c, mm	15.727	0.037	0.039
	m, g	11.457	0.018	0.018
Uncertain measurements	ρ, $\frac{g}{mm^3}$	$1.180336 \cdot 10^{-3}$	-	$8.362 \cdot 10^{-6}$
Expected result	ρ_0, $\frac{g}{mm^3}$	$1.180 \cdot 10^{-3}$	-	-

Analysis of results

Now, after our Python program counted all the results for our dice, we should analyze mathematically, whether how great result we have got, and, finally, write it in the proper way. For analysis, let's use **Student's t-test**, looks like:

$$t = \frac{\overline{\rho} - \rho_0}{U_k \rho} \tag{11}$$

So, if coefficient 't' lays in the interval (0, 1), then we can say, that our experimental results prove our theoretically expected result, and everything is alright, otherwise we should our theory or our experiment. Our program claim, that:

$$t = 0.040 \tag{12}$$

Fortunately, 't' is inside the interval and, let us say, quite near 0, that means, that we've done our experiment greatly. Last, the result:

$$\rho_{res} = 1.180 \cdot 10^{-3} \pm 8.362 \cdot 10^{-6} \frac{g}{mm^3} \tag{13}$$

4 Useful links

- Link to the git-repository with code, conditions and this article in pdf-format: https://github.com/ASTR0MNSTR/Dice-Measure.git