Quantitative Understanding in Biology
1.1 Characterizing a Distribution

Jason Banfelder

August 27th, 2024

1 Mean and Standard Deviation

Biological investigation often involves taking measurements from a sample of a population.

The mean of these measurements is, of course, the most common way to characterize their distribution:

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

The concept is easy to understand and should be familiar to everyone. However, be careful when implementing it on a computer. In particular, make sure you know how the program you are using deals with missing values.

We will be using R in this class, so let's explore how R deals with missing values. We begin by generating ten random samples from a normal distribution, and then computing the mean of those numbers:

```
x <- rnorm(10)
x

##  [1]  0.26186948 -0.24383277  1.83692046  0.66982211  0.36622646
##  [6]  1.38312991  1.27122737 -1.63038699  1.77068157 -0.07387831

mean(x)

## [1] 0.5611779
```

In R, the missing values are represented as `NA`. The meaning of `NA` is "The value exists, but I don't know what it is; it could be anything".

```
x[3] <- NA
x

## [1]  0.26186948 -0.24383277          NA  0.66982211  0.36622646
## [6]  1.38312991  1.27122737 -1.63038699  1.77068157 -0.07387831
```

With these data, the mean cannot be computed, unless you ask that missing values be ignored:

```
mean(x)

## [1] NA

mean(x, na.rm = TRUE)

## [1] 0.4194288
```

Computing the mean 'manually' requires careful attention to `NA`s:

```
sum(x)

## [1] NA

sum(x, na.rm = TRUE)

## [1] 3.774859

length(x)

## [1] 10

length(na.omit(x))

## [1] 9

sum(x, na.rm = TRUE) / length(na.omit(x))

## [1] 0.4194288
```

1.1 More on `NA`

The interpretation of `NA` in various contexts can be subtle, and not every expression that involves an `NA` will result in `NA`. For example, when performing logical `AND` (`&`) operations, if one operand is `FALSE`, it doesn't matter what the other might be:

```
FALSE & NA

## [1] FALSE
```

```
TRUE & NA

## [1] NA
```

Try to predict the output of the following expressions, which use R's logical `OR` (`|`) and `NOT` (`!`) operators, then evaluate them in R to confirm your understanding:

```
FALSE | NA
NA | TRUE
! TRUE
NA | ( ! NA)
```

1.2 Random values in R

When you run the examples above on your computer, you will generate a different set of ten (pseudo-)random numbers. While we normally would want this behavior, when testing new methods, reproducibility is very helpful. To aid in that, R allows you to set a seed from which 'random' values are generated. If you want to reproduce the exact values in the above examples, set the seed, using the `set.seed()` function, to `19710822`, and run the examples exactly as you see them here, in exactly the order that you see them.

1.3 Other measures of central tendency

In addition to the mean (or more precisely the arithmetic mean), you are probably also familiar with other measures of 'central tendency'.

The median is the value above which (and below which) 50% of the data is found. The median is less sensitive to outliers then the arithmetic mean. One case where the median can be convenient is when measuring distributions of times before an event occurs; e.g., how long it takes an animal to learn a task. If you want to report the mean, you need to wait for all the animals in your population to learn the task, which could be a really long time if there are one or two particularly dumb animals in your sample population, and may become undefined if one of your animals dies before it learns the task.[1] You can, however, report the median after just over half of your animals learn the task.

The mode is not often used in biological applications of statistics.

You may have also heard of the harmonic mean and the geometric mean. The relevant formulae are:

$$\frac{1}{H_x} = \frac{1}{n} \sum \frac{1}{x_i} \tag{2}$$

[1] If this happened to you, do you think it would be OK to represent the missing value with an `NA`?

$$GM_x = \sqrt[n]{x_1 \cdot x_2 \cdot x_3 \cdots x_n} = \sqrt[n]{\prod x_i} = e^{\frac{1}{n}\sum \ln x} \tag{3}$$

The harmonic mean is just the reciprocal of the average of the reciprocals of the measurements you have. You should think of this as transforming your data into a more natural or meaningful space, taking a regular (arithmetic) average, and then applying the inverse transform. In this case the transformation is a reciprocal.[2]

The geometric mean is similar in spirit, except that the transformation is taking a logarithm, and the inverse transformation is taking the antilog (i.e., e^x).

1.4 Measures of variation

The simplest measure of variation is the range (lowest, highest). The problem with this is that the range will typically vary systematically with sample size; we say it is a *biased* estimate. Contrast to average: your best guess of the mean of the population is the mean of the sample; thus we say the mean is an unbiased estimate of central tendency.

In addition to the mean, the standard deviation and (to a lesser extent) the variance are also commonly used to describe a distribution of values:

$$\begin{pmatrix} Sample \\ Variance \end{pmatrix} = s^2 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1} \tag{4}$$

$$\begin{pmatrix} Sample \\ Standard \\ Deviation \end{pmatrix} = s = \sqrt{\begin{pmatrix} Sample \\ Variance \end{pmatrix}} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}} \tag{5}$$

Observe that the variance is an average of the square of the distance from the mean. All terms in the summation are positive because they are squared.

When computing the variance or standard deviation (SD) of a whole population, the denominator would be n instead of $n-1$. The variance of a sample from a population is always a little bit larger, because the denominator is a little bit smaller. There are theoretical reasons for this having to do with degrees of freedom; we will chalk it up to a "weird statistics thing" (it is actually a correction that turns the SD into an unbiased estimate).

[2] In the U.S., the fuel economy of a car is reported in miles per gallon; many argue that it is more appropriate to measure fuel economy in gallons per mile (or liters per kilometer), as is done in Europe. When Congress legislates average fuel economy of the fleet of vehicles that a carmaker produces, what do you think should be averaged?

Observe that the standard deviation has the same units of measure as the values in the sample and of the mean. It gives us a measure of how spread out our data are, in units that are natural to reason with.

In the physical sciences (physics, chemistry, etc.), the primary source of variation in collected data is often due to “measurement error”: sample preparation, instrumentation, etc. This implies that if you are more careful in performing your experiments and you have better instrumentation, you can drive the variation in your data towards zero. Think about measuring the boiling point of pure water as an example. Some argue that if you need complex statistical analysis to interpret the results of such an experiment, you’ve performed the experiment badly, or you’ve done the wrong experiment.

Although one might imagine that an experimenter would always use the best possible measurement technology available (or most affordable), this is not always the case. When developing protocols for CT scans, one must consider that the measurement process can have deleterious effects on the patient due to the radiation dose required to carry out the scan. While more precise imaging, and thus measurements (say of a tumor size), can often be achieved by increasing the radiation dose, scans are selected to provide just enough resolution to make the medical diagnosis in question. In this case, better statistics means less radiation, and improved patient care.

In biological systems, the primary source of variation is often “biological diversity”. Cells and patients are rarely identical and will generally not be in identical states, so you expect a non-trivial variation, even under perfect experimental conditions. In biology, we must learn to cope with (and ultimately embrace) this naturally occurring variation.

2 Communicating a Distribution

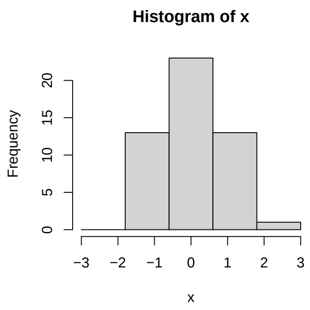
$\overline{x}$ and SD have a particular meaning when the distribution is normal. For the moment, we’ll not assume anything about normality, and consider how to represent a distribution of values.

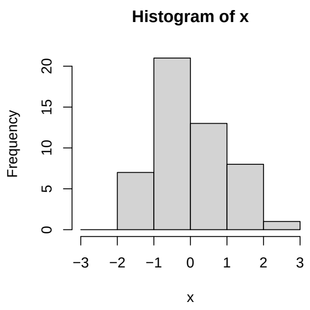
Histograms convey information about a distribution graphically. They are easy to understand, but can be problematic because binning is arbitrary. There are essentially two arbitrary parameters that you select when you prepare a histogram: the width of the bins, and the alignment, or starting location, of the bins. For non-large n, the perceptions suggested by a histogram can be misleading. To illustrate, we generate a dataset of 50 values:

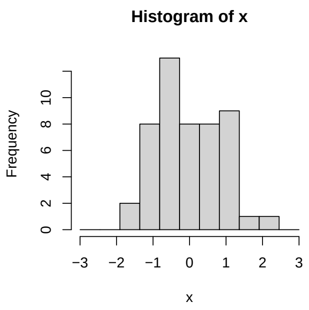
```r
set.seed(0)
x <- rnorm(50)
```

Now we will prepare three histograms, each presenting the same data; you can see that, depending on the binning, a different underlying distribution is suggested.

```
hist(x, breaks = seq(-3, 3, length.out = 6))
hist(x, breaks = seq(-3, 3, length.out = 7))
hist(x, breaks = seq(-3, 3, length.out = 12))
```



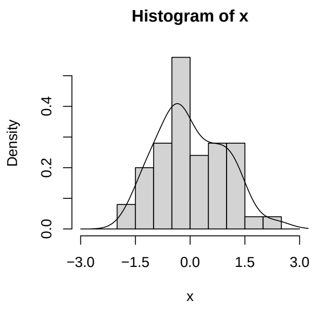




When preparing histograms, be sure that the labels on the x-axis are chosen so that the binning intervals can be easily inferred. The first plot would be better prepared by including one additional option: `xaxp = c(-3, 3, 5)`. See the entry for `par` in the R help for this and many other plotting options; type `?par` at the R prompt.

R has a less arbitrary function, `density`, which can be useful for getting the feel for the shape of an underlying distribution. This function does have one somewhat arbitrary parameter (the bandwidth); it is fairly robust and the default usually works reasonably well.

```
hist(x, breaks = seq(-3, 3, length.out = 13),
        xaxp = c(-3, 3, 4),
        probability = TRUE)
lines(density(x))
```



Note that we add the probability option to the `hist` function; this plots a normalized histogram, which is convenient, as this is the scale needed by the overlayed density function.

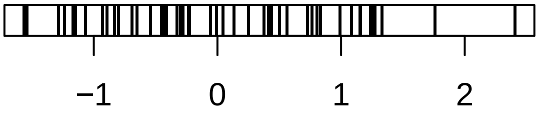
You should be wary of using summary statistics such as $\overline{x}$ and SD for samples that don't have large n or that are not known to be normally distributed. For n=50, as above, other options include:

- A table of all the values: `sort(x)`
- A more condensed version of the above: `stem(x)`

For graphical presentations, do not underestimate the power of showing all of your data. With judicious plotting choices, you can often accomplish this for n in the thousands.

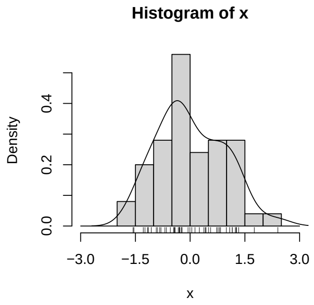
`stripchart(x)` shows all data points. For $n = 50$, you may want to set the plotting character (`pch`).

```
stripchart(x, pch = "|")
```



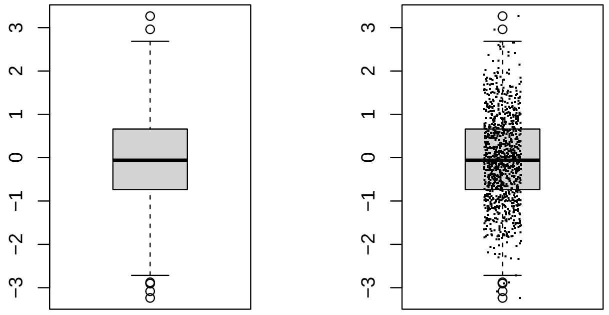
If you must prepare a histogram (it is often expected), overlaying the density curve and sneaking in a stripchart-like display can be a significant enhancement:

```
hist(x, breaks = seq(-3, 3, length.out = 13),
        xaxp = c(-3, 3, 4),
        probability = TRUE)
lines(density(x))
rug(x)
```



For larger n, a boxplot can be appropriate. You can overlay (using the `add = TRUE` option) a stripchart to show all data points. With many data points, a smaller plotting symbol and the `jitter` option are helpful.

```r
x <- rnorm(1000)
boxplot(x)
stripchart(x, vertical = TRUE, pch = ".", method = "jitter", add = TRUE)
```



Note that boxplots show quartiles. The heavy bar in the middle is the median, not the mean. The box above the median is the third quartile; 25% of the data falls in it. Similarly, the box below the median holds the second quartile. The whiskers are chosen such that, if the underlying distribution is normal, roughly 1 in 100 data points will fall outside their range. These are putative outliers that you may want to inspect further.

The concept of quartiles can be generalized to quantiles; quartiles and deciles are favorites:

```
quantile(x, (0:4) / 4)
```

```
##          0%         25%         50%         75%        100%
## -3.23638573 -0.73450676 -0.06082413  0.66041946  3.26641452
```

```
quantile(x, (0:10) / 10)
```

```
##          0%         10%         20%         30%         40%         50%
## -3.23638573 -1.34829984 -0.93057704 -0.58140841 -0.28630903 -0.06082413
##         60%         70%         80%         90%        100%
##  0.19523786  0.51246497  0.84763947  1.29953657  3.26641452
```

SD is a representation of how spread out your data are. If the underlying distribution is normal and n is large, then 95% of the samples are expected to fall within the range: $\overline{x} \pm 1.96 \cdot SD$.

```
x <- rnorm(100000)
c(mean = mean(x), sd = sd(x))
```

```
##        mean          sd
## 0.001697208 1.000869350
```

```
quantile(x, (0:40) / 40)
```

```
##           0%         2.5%           5%         7.5%          10%
## -4.471030759 -1.965430815 -1.650145353 -1.442248402 -1.283966354
##        12.5%          15%        17.5%          20%        22.5%
## -1.148979951 -1.037947503 -0.935441703 -0.841484539 -0.754917702
##          25%        27.5%          30%        32.5%          35%
## -0.678286586 -0.600782370 -0.525201710 -0.452817481 -0.382528301
##        37.5%          40%        42.5%          45%        47.5%
## -0.315102862 -0.250841314 -0.185004156 -0.120821554 -0.056733457
##          50%        52.5%          55%        57.5%          60%
##  0.005256214  0.067676610  0.130156544  0.192932958  0.257288815
##        62.5%          65%        67.5%          70%        72.5%
##  0.322367352  0.389814605  0.458908876  0.530046282  0.602453978
##          75%        77.5%          80%        82.5%          85%
##  0.678796796  0.760053413  0.843131781  0.934117988  1.037211767
##        87.5%          90%        92.5%          95%        97.5%
##  1.147183645  1.278740328  1.437153591  1.638732858  1.959821449
##         100%
##  4.336132109
```

We expect the mean to be zero, the SD to be unity, the 2.5% quantile to be at -1.96, and the 97.5% quantile to be at +1.96.

3 Standard Deviation vs. Standard Error of the Mean

An important, but very different, question that statistics can help us with is how well we can estimate the mean. Two factors influence this: how spread out the data are, and how much data we have. A new quantity, the Standard Error of the Mean, is introduced:

$$SEM = \frac{SD}{\sqrt{n}} \tag{6}$$

For large n, we can be 95% sure that the true mean of the underlying population is in the range. . .

$$\overline{x} \pm 1.96 \cdot SEM \tag{7}$$

. . . where $\overline{x}$ is the sample mean. We will formalize and extend this result in another session.

Here is an experiment to demonstrate this. We generate a sample from a known normal distribution where the mean is zero and the standard deviation is unity, then compute a confidence interval (CI) for the mean. We expect that this CI will contain the true mean (which we know to be zero) roughly 19 out of 20 times.

```
set.seed(0)
num_trials <- 100
d <- data.frame(lower = numeric(num_trials), mean = numeric(num_trials), upper = numeric(num_trials))
for (i in 1:num_trials) {
  x <- rnorm(10000)
  d[i, ] <- mean(x) + c(-1.96, 0, +1.96) * sd(x) / sqrt(length(x))
}
d
```

```
##             lower          mean         upper
## 1   -9.078317e-03  0.0104475229  0.0299733625
## 2   -2.415354e-02 -0.0046166370  0.0149202662
## 3   -1.292448e-02  0.0069563198  0.0268371225
## 4   -1.833335e-02  0.0014819895  0.0212973304
## 5   -3.423455e-02 -0.0146767588  0.0048810344
## 6   -2.290620e-02 -0.0035945569  0.0157170908
## 7   -7.200773e-03  0.0124330418  0.0320668570
## 8   -1.860558e-02  0.0011154055  0.0208363931
## 9   -2.553421e-02 -0.0058048453  0.0139245237
## 10  -9.129604e-03  0.0103549932  0.0298395904
## 11  -1.652821e-02  0.0030184302  0.0225650703
## 12  -1.781742e-02  0.0015974985  0.0210124196
## 13  -2.767718e-02 -0.0080991131  0.0114789512
## 14  -3.163399e-02 -0.0120506457  0.0075326938
## 15  -1.497326e-02  0.0046855740  0.0243444112
## 16  -3.192447e-02 -0.0124209527  0.0070825611
## 17  -4.869262e-02 -0.0290074870 -0.0093223523
## 18  -2.391941e-02 -0.0045064895  0.0149064263
## 19  -2.364406e-02 -0.0041236286  0.0153968013
## 20  -1.163170e-03  0.0183558682  0.0378749069
## 21  -1.704554e-02  0.0024607619  0.0219670615
## 22  -2.697047e-02 -0.0074509234  0.0120686254
## 23  -3.058368e-02 -0.0109863196  0.0086110372
## 24  -3.629121e-02 -0.0167812965  0.0027286152
## 25  -6.457957e-03  0.0132738887  0.0330057348
## 26  -1.327635e-02  0.0063118169  0.0258999805
## 27  -2.517593e-02 -0.0057416876  0.0136925549
## 28  -3.541489e-03  0.0160433649  0.0356282186
## 29  -1.620469e-02  0.0034256925  0.0230560706
## 30  -2.810554e-02 -0.0083895415  0.0113264575
## 31  -1.612765e-02  0.0036372419  0.0234021313
## 32  -2.612756e-02 -0.0066477882  0.0128319837
## 33  -4.560199e-03  0.0151358045  0.0348318083
## 34  -1.744546e-02  0.0019528221  0.0213511007
## 35  -2.163522e-02 -0.0021430419  0.0173491409
## 36  -7.881836e-03  0.0119868922  0.0318556209
## 37  -2.529271e-03  0.0173170587  0.0371633884
## 38  -2.424531e-02 -0.0048472456  0.0145508139
## 39  -2.848581e-02 -0.0090926365  0.0103005328
## 40   4.021239e-03  0.0235616292  0.0431020194
## 41  -1.974404e-02 -0.0001320712  0.0194798942
## 42  -1.421975e-02  0.0052335464  0.0246868411
## 43  -1.232026e-02  0.0071075977  0.0265354597
## 44  -1.616949e-03  0.0178620649  0.0373410782
## 45  -1.446737e-02  0.0050133735  0.0244941172
## 46  -4.248768e-02 -0.0228392697 -0.0031908565
## 47  -6.277747e-03  0.0132894309  0.0328566084
## 48  -2.440202e-02 -0.0048483767  0.0147052661
## 49  -4.347561e-02 -0.0238829801 -0.0042903481
## 50  -2.349321e-02 -0.0040054422  0.0154823293
## 51  -1.903669e-02  0.0006025682  0.0202418278
## 52  -1.947187e-02  0.0003092677  0.0200904052
## 53  -1.745579e-02  0.0021978746  0.0218515387
## 54  -2.716072e-02 -0.0077055227  0.0117496781
## 55  -1.370284e-02  0.0059190322  0.0255409009
## 56  -5.151753e-03  0.0144996573  0.0341510680
## 57  -3.098236e-02 -0.0113657093  0.0082509449
## 58  -1.689180e-02  0.0027974273  0.0224866498
## 59   7.696701e-03  0.0272560912  0.0468154813
## 60  -1.412135e-02  0.0055954918  0.0253123373
## 61  -9.562056e-03  0.0104354511  0.0304329586
## 62  -1.615755e-02  0.0034056409  0.0229688302
## 63  -1.398204e-02  0.0055182010  0.0250184401
## 64  -2.593795e-02 -0.0062158068  0.0135063413
## 65  -1.893294e-02  0.0007378252  0.0204085865
## 66  -2.538459e-02 -0.0057889154  0.0138067578
## 67  -2.485491e-02 -0.0051136874  0.0146275353
## 68  -2.583081e-02 -0.0063699715  0.0130908690
## 69  -1.616647e-02  0.0033567188  0.0228799115
## 70  -1.672108e-02  0.0029577350  0.0226365509
## 71  -1.813827e-02  0.0015369098  0.0212120913
## 72  -1.258386e-02  0.0070408650  0.0266655860
## 73  -1.366018e-02  0.0058652798  0.0253907412
## 74  -1.881063e-02  0.0007527620  0.0203161493
## 75   2.343312e-05  0.0196816604  0.0393398877
## 76  -3.276566e-02 -0.0131579721  0.0064497156
## 77  -2.937736e-02 -0.0096678060  0.0100417467
## 78  -2.324920e-02 -0.0037669304  0.0157153426
## 79  -4.258805e-02 -0.0229237199 -0.0032593865
## 80  -2.538248e-02 -0.0057483725  0.0138857396
## 81  -2.489160e-02 -0.0052718073  0.0143479861
## 82  -2.310554e-02 -0.0034134214  0.0162786944
## 83  -2.394009e-02 -0.0042738337  0.0153924225
## 84  -1.195651e-02  0.0077363680  0.0274292433
## 85  -2.141509e-02 -0.0020009653  0.0174131630
## 86  -9.003688e-03  0.0106698340  0.0303433557
## 87  -2.086674e-02 -0.0013727595  0.0181212227
## 88  -1.367983e-02  0.0060671468  0.0258141282
## 89  -3.538666e-02 -0.0156031419  0.0041803759
## 90  -1.555023e-02  0.0040712904  0.0236928156
## 91  -2.485455e-02 -0.0052911167  0.0142723189
## 92  -3.836672e-02 -0.0188405885  0.0006855389
## 93  -3.230642e-02 -0.0129695653  0.0063672938
## 94  -2.406534e-02 -0.0045094168  0.0150465075
## 95  -2.118849e-02 -0.0015383105  0.0181118711
## 96  -2.261500e-02 -0.0030566793  0.0165016406
## 97  -1.181274e-02  0.0077905620  0.0273938593
## 98  -2.403643e-02 -0.0045491751  0.0149380749
## 99  -1.108083e-03  0.0182848112  0.0376777051
## 100 -2.525228e-02 -0.0056995141  0.0138532547
```

```
(bad_estimate_count <- length(which(d$lower > 0 | d$upper < 0)))
```

```
## [1] 7
```

This is pretty close to what was expected; in this particular experiment, the true mean was not within the CI in 7 cases out of 100 (we expected about five).

Understanding the difference between the SD and the SEM is critical. To reiterate, the SD gives us an indication of how spread out the data in the underlying population are. The SEM is an indication of how confident we are in our estimate of the true mean of the underlying population.

Many plots in publications show error bars. There is no standard as to what these represent; it could be $\pm SD$, $\pm SEM$, $\pm 1.96 \cdot SD$, $\pm 1.96 \cdot SEM$, or, as we will see later, something else. If the publication does not explicitly state what the error bars represent, they are of no use to you (and you might begin to question the underlying analysis).

4 Homework

Although there is nothing to hand in for this homework, it is required and the material herein may appear on exams.

- Read Cumming, Fidler, and Vaux. Error bars in experimental biology. *J Cell Biol.* 2007 Apr 9; 177(1): 7-11.
- Install R (`https://www.r-project.org/`) and R Studio (`https://www.rstudio.com/`) on your computer. Make sure you can run all of the examples in these notes. If you get tired of typing some of the longer examples, all of the R commands are available from the course website.
- There are many excellent books, tutorials, and other resources for learning about R; one that we recommend is the `swirl` package. Install `swirl` like this...

  ```
  install.packages("swirl")
  ```

  Then, to start the tutorial...

  ```
  library(swirl)
  swirl()
  ```