Problem set 4: Worked solutions

Statistics and statistical programming
Northwestern University
MTS 525

Aaron Shaw

October 19, 2020

Contents

Programming Challenges

PC1. Import

Load the dataset. As usual, you'll need to edit this block of code to point at the appropriate file location to make it work on your machine.

```r
pop <- read.delim(file = url("https://communitydata.science/~ads/teaching/2020/stats/data/week_06/popul

# Same thing using `read_tsv()`. Notice that it can handle URLs directly.
library(tidyverse)

pop <- read_tsv("https://communitydata.science/~ads/teaching/2020/stats/data/week_06/population.tsv")

head(pop)
```

```
## # A tibble: 6 x 6
##        x     j     l     k     y group
##    <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
## 1  0.200     1     1     1  1.1  group_01
## 2 NA         0     1     2 NA    group_01
## 3  0.164     1     0     3 -3.01 group_01
## 4  3.50      1     1     1 11    group_01
## 5  0.183     1     0     3  2.55 group_01
## 6  7.57      0     1     3 21.7  group_01
```

While we're at it, let's import the data from Problem Set 2:

```
## Insert an appropriate group number as needed:
ps2 <- read.csv(file = url("https://communitydata.science/~ads/teaching/2020/stats/data/week_04/group_03

head(ps2)
```

```
##          x j l k     y
## 1 1.794517 1 0 1  9.38
## 2 0.572927 1 0 2  4.72
## 3 3.256708 0 1 1 14.27
## 4 1.009964 0 1 1  6.03
## 5 0.744927 0 1 1 -2.77
## 6 6.120917 1 0 2 17.86
```

PC2. Compare means

Straightforward enough:

```
mean(pop$x, na.rm = T)
```

```
## [1] 2.595362
```

```
mean(ps2$x, na.rm = T)
```

```
## [1] 2.845581
```

Given the structure of the dataset for this week, it's also no big deal to calculate all of the group means (and you can compare your answers against these if you like). Notice that I can define and call a function within my call to `tapply()`:

```
s.means <- tapply(pop$x, pop$group, function(x) {
  round(mean(x, na.rm = T), 2)
})

s.means
```

```
## group_01 group_02 group_03 group_04 group_05 group_06 group_07 group_08
##     2.97     3.10     2.85     2.12     2.72     2.62     2.78     2.09
## group_09 group_10 group_11 group_12 group_13 group_14 group_15 group_16
##     2.84     2.30     2.26     2.55     2.33     2.49     2.71     2.61
## group_17 group_18 group_19 group_20
##     2.69     2.56     2.57     2.74
```

For anyone who decided to try this using a Tidyverse approach, here's a way you might do that with a call to `group_by()` and `summarize()`:

```
pop %>%
  group_by(group) %>%
  summarize(
    mean(x, na.rm = T)
  )
```

```
## # A tibble: 20 x 2
##    group    `mean(x, na.rm = T)`
##    <chr>                   <dbl>
##  1 group_01                 2.97
##  2 group_02                 3.10
##  3 group_03                 2.85
##  4 group_04                 2.12
##  5 group_05                 2.72
##  6 group_06                 2.62
##  7 group_07                 2.78
##  8 group_08                 2.09
##  9 group_09                 2.84
## 10 group_10                 2.30
## 11 group_11                 2.26
## 12 group_12                 2.55
## 13 group_13                 2.33
## 14 group_14                 2.49
## 15 group_15                 2.71
## 16 group_16                 2.61
## 17 group_17                 2.69
## 18 group_18                 2.56
## 19 group_19                 2.57
## 20 group_20                 2.74
```

Knowing that each group was a random sample from the entire population, we might think of the individual group (sample) means as constructing a *sampling distribution of the (population) mean.*

PC3. CI of the mean

I'll do this two ways. First, I can plug in the values from one group sample into the formula for the standard error and then add/subtract twice the standard error from the mean to find the 95% CI.

```r
se <- sd(ps2$x, na.rm = T) / sqrt(length(ps2$x[!is.na(ps2$x)]))

mean(ps2$x, na.rm = T) - (2 * se) ## lower
```

```
## [1] 2.298214
```

```r
mean(ps2$x, na.rm = T) + (2 * se) ## upper
```

```
## [1] 3.392948
```

Now, I'll write a more general function to calculate confidence intervals. Note that my function here takes an argument for `x` as well as an `alpha` argument with a default value of 0.05. The function then goes on to divide alpha (and it's complementary probability) by 2. Can you explain why this division step is necessary?

```r
ci <- function(x, alpha = 0.05) {
  x <- x[!is.na(x)]
  probs <- c(alpha / 2, 1 - alpha / 2)
  critical.values <- qnorm(probs, mean = 0, sd = 1)
  se <- sd(x) / sqrt(length(x))
  return(
    round(mean(x) + critical.values * se, 2)
  )
}

## Here I run the function on the group 3 data
ci(ps2$x)
```

```
## [1] 2.31 3.38
```

Again, it's possible to use `tapply()` to estimate this for every group:

```r
group.confints <- tapply(pop$x, pop$group, ci)
group.confints
```

```
## $group_01
## [1] 2.46 3.48
##
## $group_02
## [1] 2.58 3.62
##
## $group_03
## [1] 2.31 3.38
##
## $group_04
## [1] 1.71 2.52
##
## $group_05
## [1] 2.25 3.20
##
## $group_06
## [1] 2.19 3.06
##
## $group_07
## [1] 2.29 3.27
##
## $group_08
## [1] 1.73 2.46
##
## $group_09
## [1] 2.30 3.38
##
## $group_10
## [1] 1.95 2.64
##
## $group_11
## [1] 1.85 2.67
##
## $group_12
## [1] 2.10 2.99
##
## $group_13
## [1] 1.92 2.74
##
## $group_14
## [1] 2.08 2.90
##
## $group_15
## [1] 2.28 3.14
##
## $group_16
## [1] 2.17 3.05
##
## $group_17
## [1] 2.26 3.13
##
```

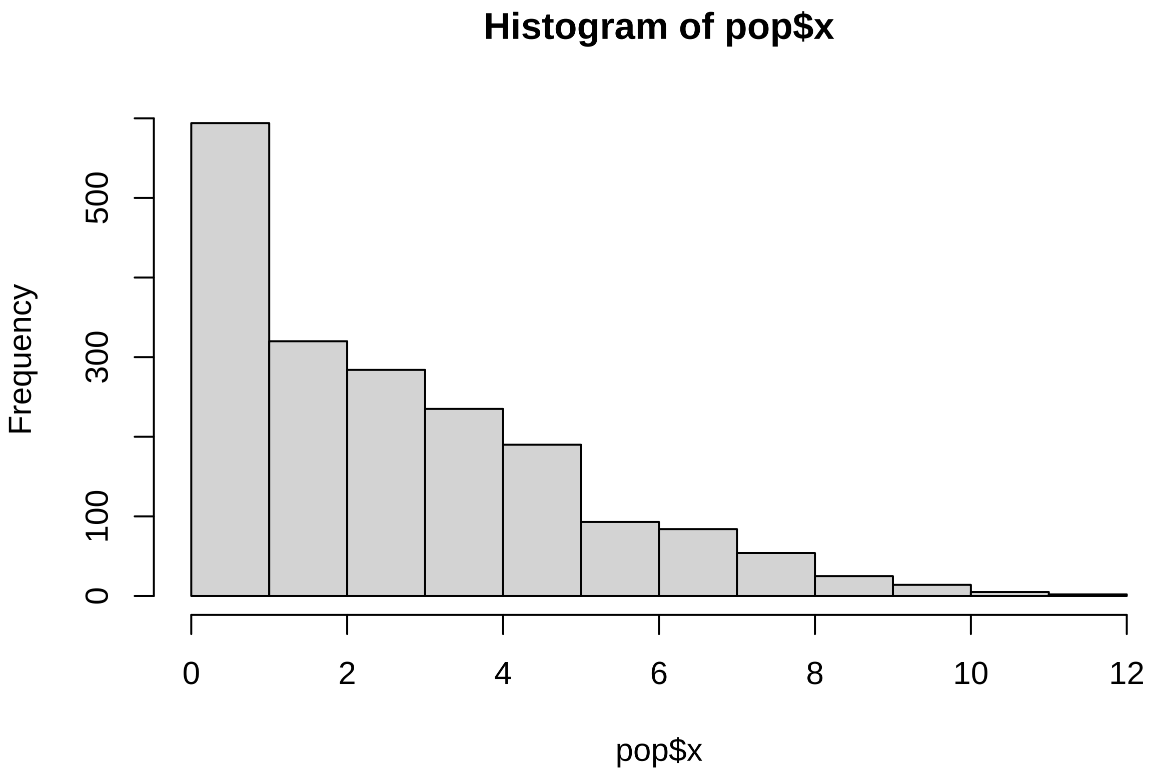
```
## $group_18
## [1] 2.08 3.04
##
## $group_19
## [1] 2.12 3.03
##
## $group_20
## [1] 2.31 3.17
```

Recall that the population mean for `x` was 2.60. Since the group samples are random samples, we should not be surprised that the group means are different from the population mean. We should also not be surprised that the 95% CI for the population mean estimated from at least one of the samples does *not* include the true population mean. Recall that our confidence interval is 95%, so we can expect to be wrong about 1/20 times (on average)! In this case, we got unlucky since 2 of our confidence intervals around the sample means do not include the population mean.

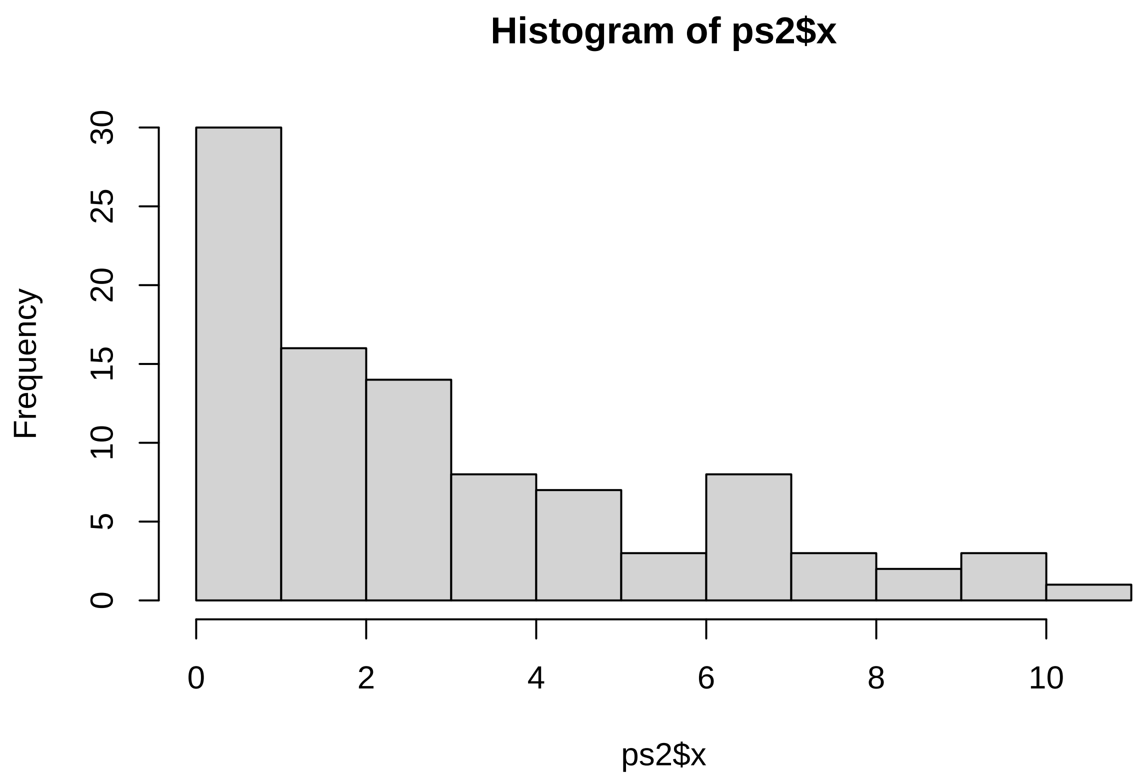
PC4. Compare distributions

I'll start with the single comparison between the population and my Problem Set 2 sample using base-R functions:

```r
hist(pop$x)
```



```r
hist(ps2$x)
```



```r
summary(pop$x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##  0.0000  0.6662  2.1540  2.5954  3.9514 11.2186     100
```

```r
sd(pop$x, na.rm = T)
```

```
## [1] 2.248925
```

```r
summary(ps2$x)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's 
##  0.001016  0.713532  2.098321  2.845581  4.580239 10.425881         5
```

```r
sd(ps2$x, na.rm = T)
```

```
## [1] 2.667537
```

Notice the differences between the shapes of the histograms as well as the summary statistics. In particular, you might consider that the sample has a *larger* standard deviation than the population. Given what you know about this statistic and the conditions under which this data was generated, could it have worked out otherwise (i.e., could the standard deviation of the population have been larger than that of the sample?). How?

Moving right along, here's `tapply()` code to construct the same comparisons for each group.

```r
tapply(pop$x, pop$group, summary)
```

```
## $group_01
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's 
## 0.000836 0.687598 2.468424 2.969925 4.752613 9.229073        5 
## 
## $group_02
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.07774 0.98868 2.15836 3.10004 4.58581 9.55063       5
##
## $group_03
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.001016  0.713532  2.098321  2.845581  4.580239 10.425881         5
##
## $group_04
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
##  0.01824  0.41583  1.57284  2.11893  3.20552 10.23085        5
##
## $group_05
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.009661  0.749035  2.046456  2.724747  4.254075 10.501638         5
##
## $group_06
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.001142 0.609075 2.296833 2.623466 3.793058 8.225825        5
##
## $group_07
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.000002 0.771873 2.229584 2.784415 4.378727 9.664410        5
##
## $group_08
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.003612 0.483745 1.813343 2.093355 3.230280 8.230183        5
##
## $group_09
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.007371  0.391076  2.452967  2.837696  4.146285 11.142319         5
##
## $group_10
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.008448 0.867873 2.010561 2.295674 3.218866 7.758600        5
##
## $group_11
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.000406 0.483762 1.974931 2.262565 3.245490 8.876453        5
##
## $group_12
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.001617 0.679982 1.880070 2.545514 4.302193 9.086439        5
##
## $group_13
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.00668 0.52002 1.96659 2.33019 3.67882 9.05578       5
##
## $group_14
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.001154 0.817183 2.127600 2.488783 3.766750 7.420036        5
##
## $group_15
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.008581 0.889763 2.407130 2.713708 4.225709 8.307670        5
##
```

```
## $group_16
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.008074  0.693095  2.498012  2.608808  4.091852 10.052089         5
##
## $group_17
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.01861 0.77353 2.42977 2.69254 4.01225 8.36492       5
##
## $group_18
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.001341  0.443132  1.934615  2.558555  3.924002 11.218583         5
##
## $group_19
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##  0.003978  0.798635  2.138282  2.571368  3.721506 10.499781         5
##
## $group_20
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.008241 0.902560 2.257811 2.741392 4.005641 8.206326        5
```

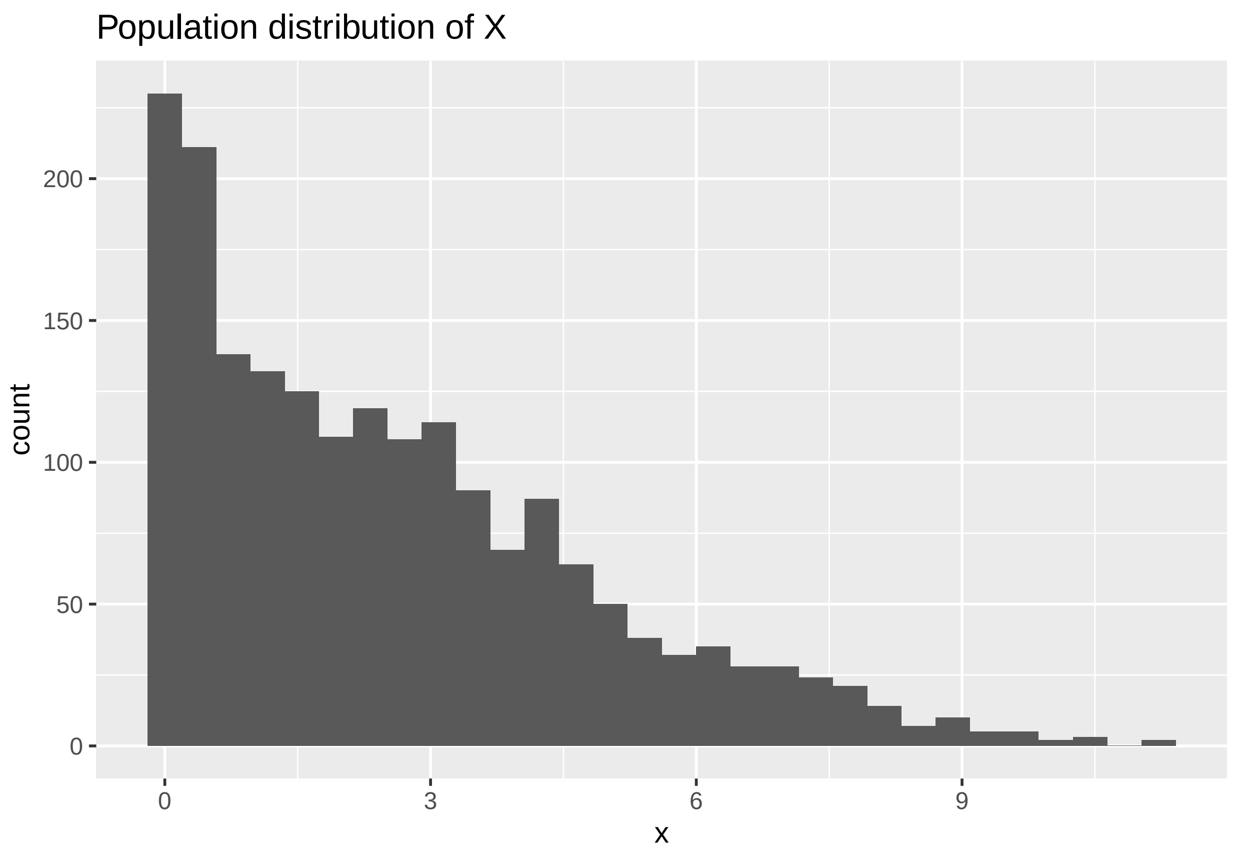
```
tapply(pop$x, pop$group, sd, na.rm = T)
```

```
## group_01 group_02 group_03 group_04 group_05 group_06 group_07 group_08
## 2.517224 2.578032 2.667537 2.014241 2.384387 2.174388 2.438099 1.819085
## group_09 group_10 group_11 group_12 group_13 group_14 group_15 group_16
## 2.678135 1.705626 2.036020 2.198885 2.040294 2.053329 2.142014 2.170530
## group_17 group_18 group_19 group_20
## 2.167427 2.401010 2.267460 2.148037
```

And here's a visual comparison. A ggplot2 "faceted" set of histograms will produce an easy visual comparison (and also makes for concise code). First I'll do it in a slightly simplified way:

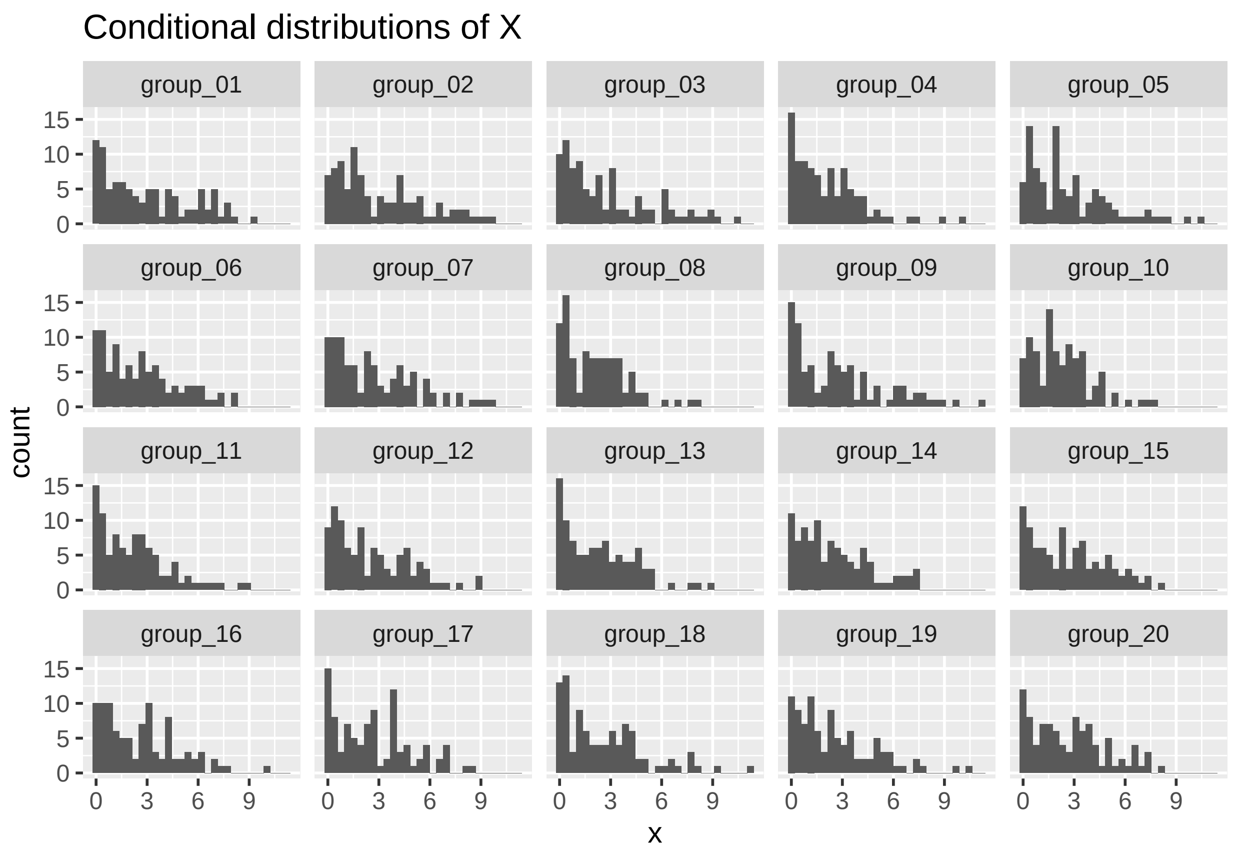
```
library(ggplot2)
ggplot(pop, aes(x)) +
  geom_histogram() +
  labs(title = "Population distribution of X")
```

```
## Warning: Removed 100 rows containing non-finite values (stat_bin).
```



```r
ggplot(pop, aes(x)) +
  geom_histogram() +
  facet_wrap(. ~ group) +
  labs(title = "Conditional distributions of X")
```

```
## Warning: Removed 100 rows containing non-finite values (stat_bin).
```

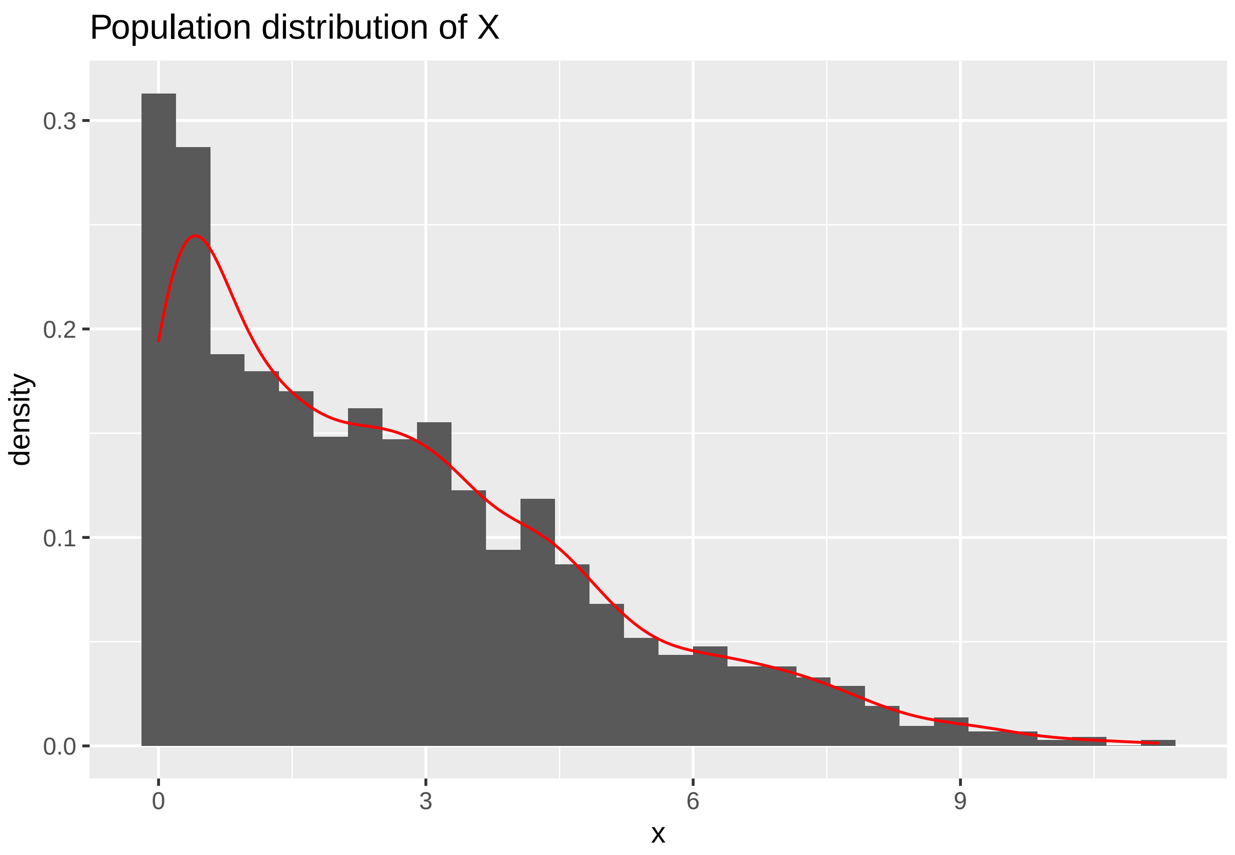


It's possible to see some of the differences there, but it might be helpful to overlay a brightly colored density plot. In the process, I can also convert the histograms themselves into density plots to make it easier to directly contrast each group against the population without having to worry about the divergent y-axis scales. The code below does that by providing a call to `after_stat(density)` inside the initial plotting aesthetics and a `color` argument to `geom_density()`. For more examples like this, you might search the phrase "density histogram" online.

```r
library(ggplot2)
ggplot(pop, aes(x, after_stat(density))) +
  geom_histogram() +
  geom_density(color = "red") +
  labs(title = "Population distribution of X")
```

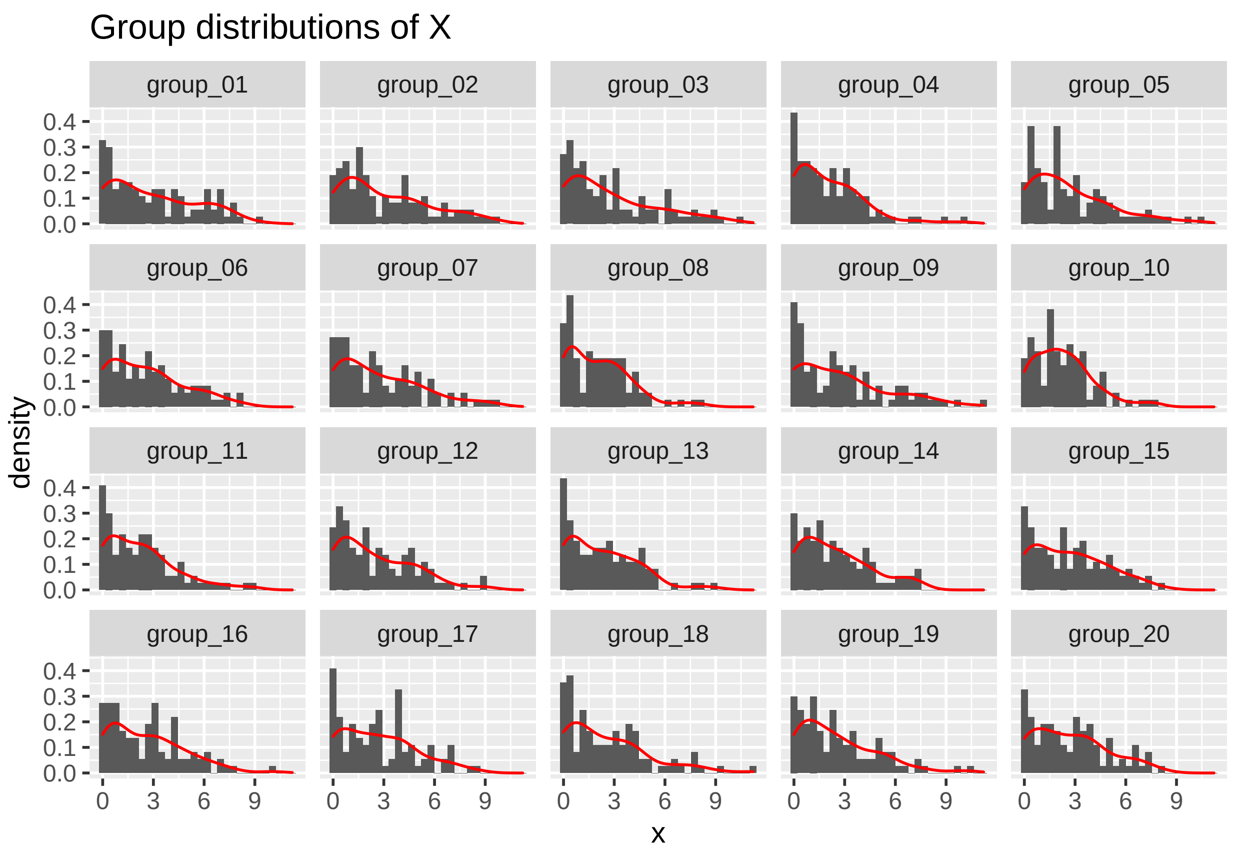
```
## Warning: Removed 100 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 100 rows containing non-finite values (stat_density).
```



```
ggplot(pop, aes(x, after_stat(density))) +
  geom_histogram(aes(x, )) +
  facet_wrap(. ~ group) +
  geom_density(color = "red") +
  labs(title = "Group distributions of X")
```

```
## Warning: Removed 100 rows containing non-finite values (stat_bin).

## Warning: Removed 100 rows containing non-finite values (stat_density).
```



The conditional distributions all look a little bit different from each other and from the population distribution. Again, none of this should be shocking given the relationship of the samples to the population.

PC5. Std. dev. of conditional means

I can do this by constructing my vector of group means again and calling `sd()` on that.

```
s.means <- tapply(pop$x, pop$group, mean, na.rm = T)
s.means
```

```
## group_01 group_02 group_03 group_04 group_05 group_06 group_07 group_08
## 2.969925 3.100038 2.845581 2.118926 2.724747 2.623466 2.784415 2.093355
## group_09 group_10 group_11 group_12 group_13 group_14 group_15 group_16
## 2.837696 2.295674 2.262565 2.545514 2.330186 2.488783 2.713708 2.608808
## group_17 group_18 group_19 group_20
## 2.692536 2.558555 2.571368 2.741392
```

```
sd(s.means)
```

```
## [1] 0.2695238
```

```
## This was the standard error from one of the groups that I calculated earlier:
se
```

```
## [1] 0.2736836
```

As mentioned earlier, the distribution of sample means drawn from the population is the *sampling distribution*. The standard error of the mean estimated from any of the individual groups/samples should be a good approximation of (but not necessarily equal to!) the standard deviation of the sampling distribution of the means.

PC6. A simulation

Since there's going to be some randomness in the next few commands, I'll set a seed value to ensure the results are reproducible and consistent on other machines.

```
set.seed(20201019)
```

(a) Simulate draws from a known distribution

There are a few ways to do this, but one of the most intuitive is to define my vector of possible values and then sample it repeatedly *with replacement.*

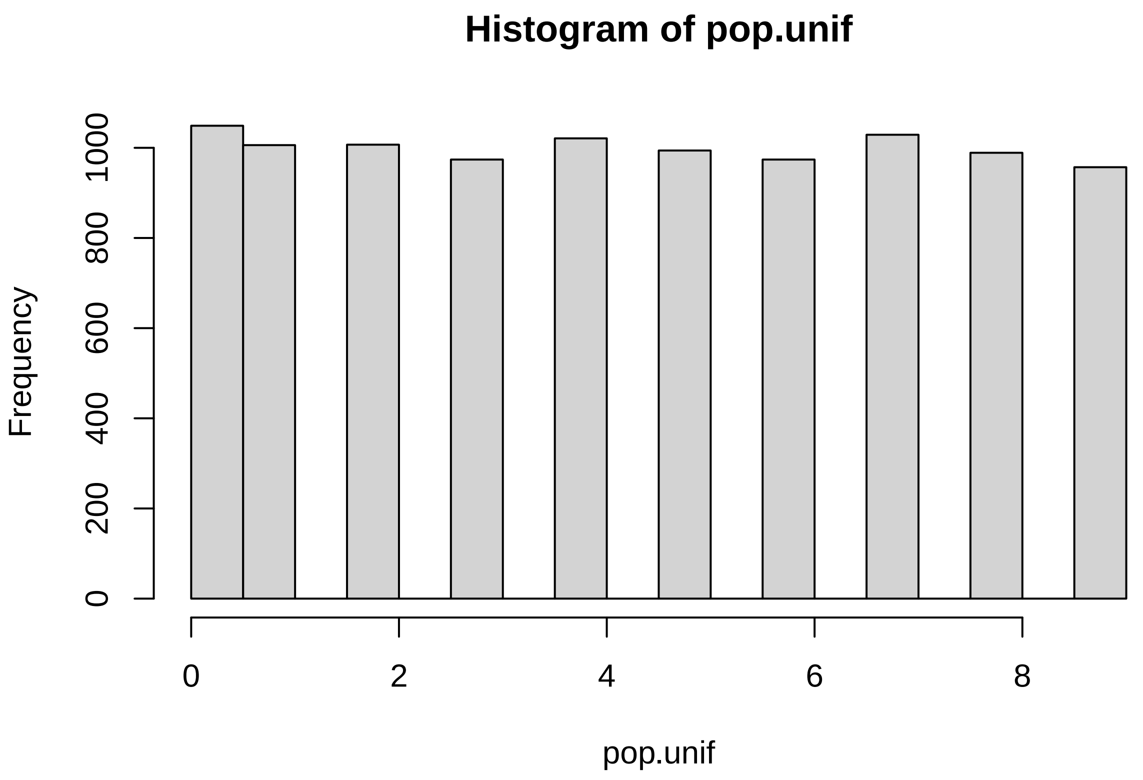
```
my.vals <- seq(0, 9)
pop.unif <- sample(my.vals, 10000, replace = TRUE)
```

(b) Take the mean

```
mean(pop.unif)
```

```
## [1] 4.4568
```

```
hist(pop.unif)
```



(c) Draw samples and describe them

Again, many ways to go about this. A good first step might be to do the sampling part once:

```
sample(pop.unif, 2, replace = T)
```

```
## [1] 7 9
```

Now I can call the mean of that:

```
mean(sample(pop.unif, 2, replace = T))
```

```
## [1] 2
```

Now, I want to run that 100 times. I might do that with a for-loop and store the values in a new vector:

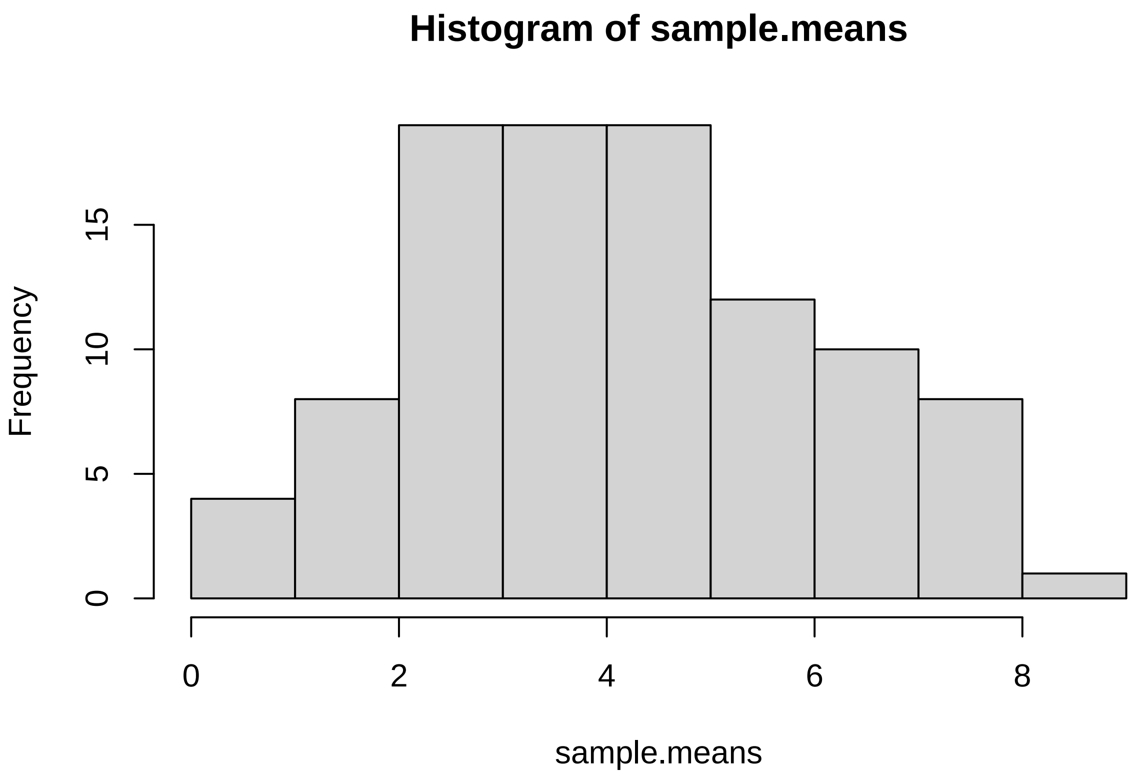
```r
sample.means <- 0

for (i in 1:100) {
  sample.means[i] <- mean(sample(pop.unif, 2, replace = T))
}
```

You could also do the same thing by nesting the sampling step inside of a call to `sapply()` that runs over an arbitrary index (`rep(1, 100)` here).

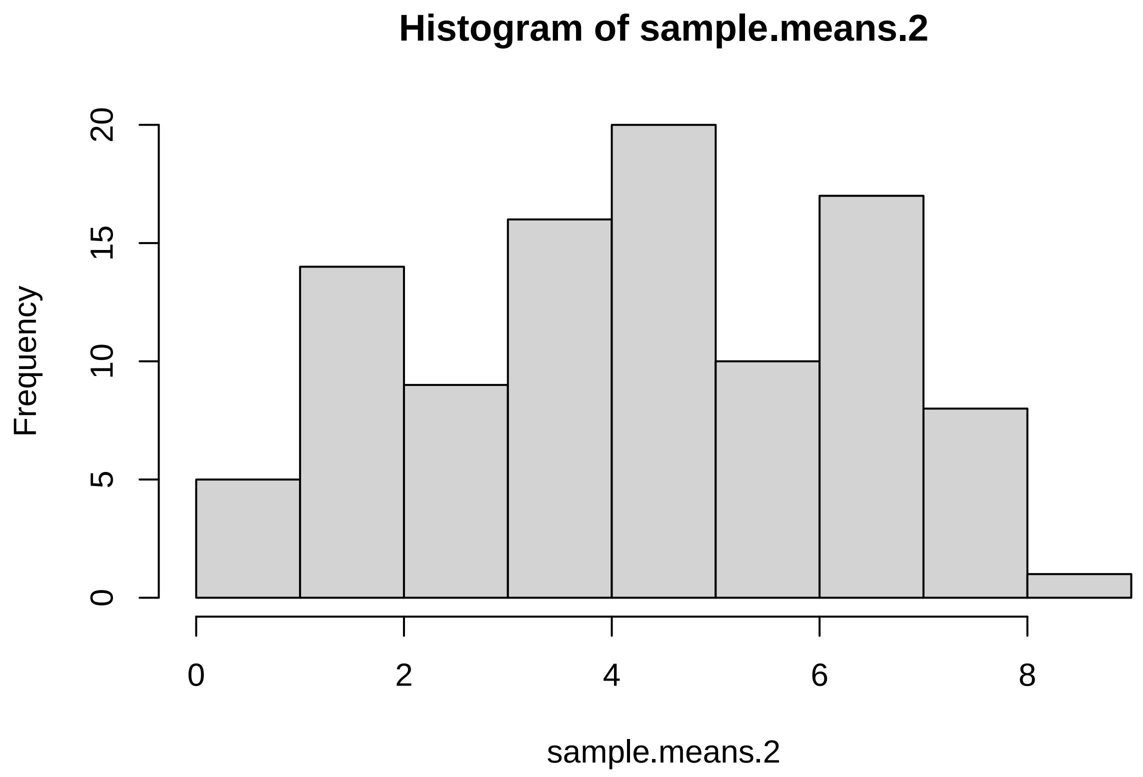
```r
sample.means.2 <- sapply(rep(1, 100), function(x) {
  mean(sample(pop.unif, 2))
})
```

Then I can plot them:

```r
hist(sample.means)
```



```r
hist(sample.means.2)
```

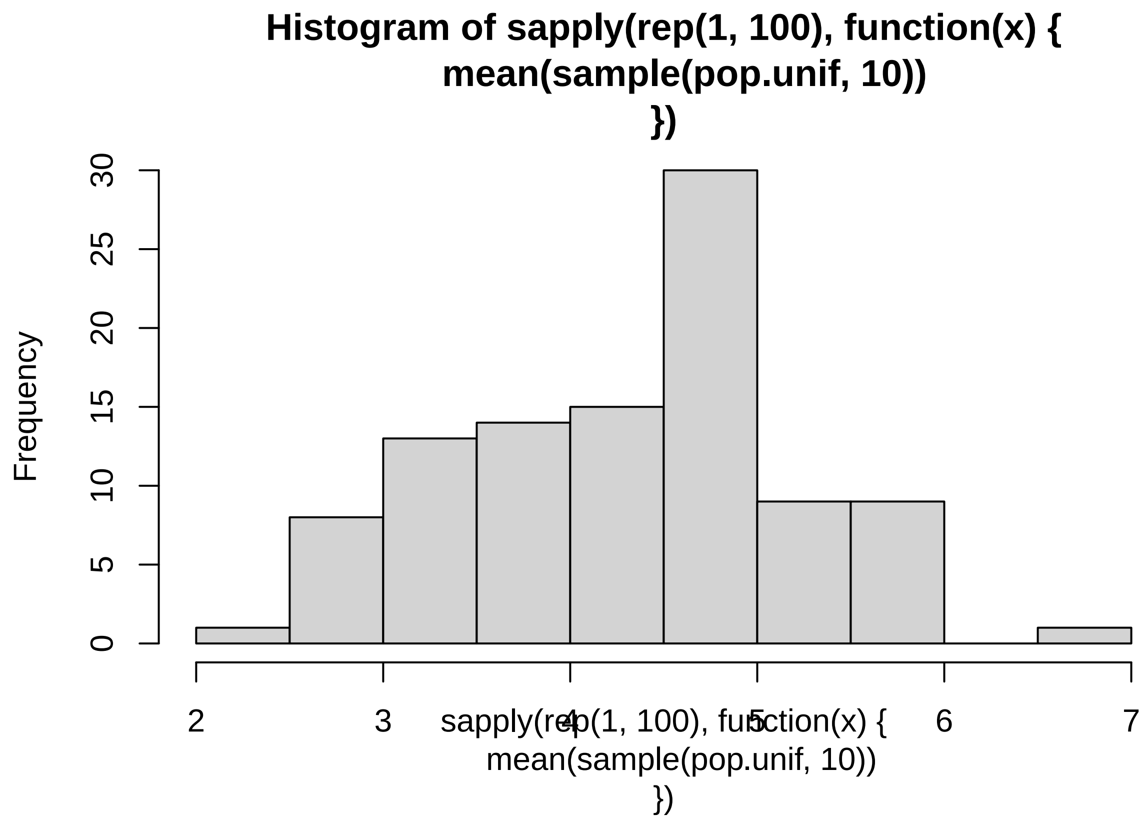


Note that you certainly didn't need to do this twice, but having done so provides a nice illustration of sampling variability!

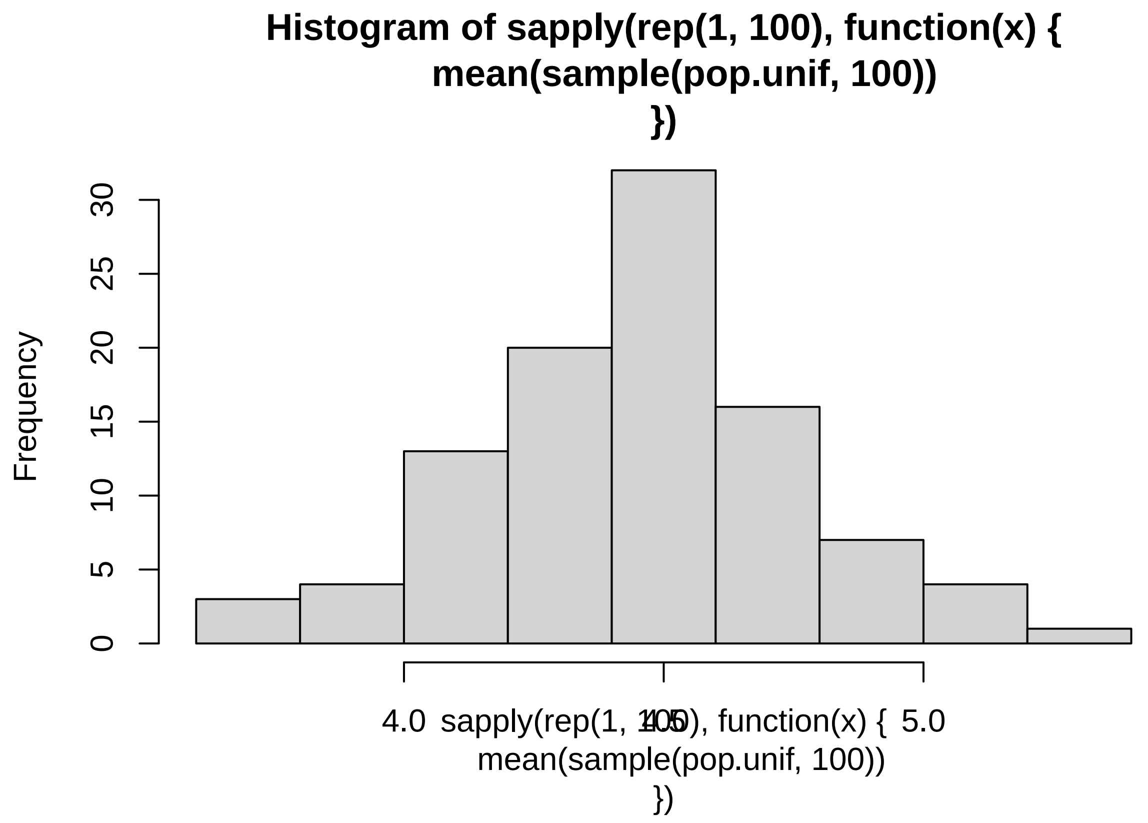
(d) Draw more samples and describe them too

Same `tapply()` command as (c) just changing the sample size

```r
hist(sapply(rep(1, 100), function(x) {
  mean(sample(pop.unif, 10))
}))
```



```r
hist(sapply(rep(1, 100), function(x) {
  mean(sample(pop.unif, 100))
}))
```



Those axis labels are just terrible, but let's try to focus on the substance of the plots. Some notable things you might observe include that the sampling distribution of the means approaches normality as each sample gets larger in size (and this is true whether the population we draw from is uniform, log-normal, or really just about any other smooth distribution). In this simulation, the number of samples is constant (100), so the changes in the distribution are *solely* due to changes in sample size. This is an illustration of some aspects of the *central limit theorem* applied to sampling distributions. It is also an illustration of the *t-distribution* (the basis for "t-tests" that you'll read about soon).

Reading Questions

RQ1. CIs vs. P-values?

We'll discuss this one as a group and I'm eager to hear others' thoughts. Personally, I am inclined to agree with Reinhart as I find the focus on p-values somewhat thought-stopping and would prefer to see researchers and publications embrace reporting standards that yield more intuitive, flexible, uncertain, and meaningful interpretations of findings in terms of the original measurements/units. Confidence intervals usually accomplish these goals more effectively than p-values and, in that respect, I like confidence intervals quite a lot.

RQ2 Emotional contagion revisited

(a) Hypotheses

In my words (or rather formulas since I think that's less ambiguous), the key pairs of null/alternative hypotheses look something like the following:

Let Δ be the parameter estimate for the difference in mean percentage of positive (μ_{pos}) and negative (μ_{neg}) words between the experimental and control conditions for the treatments of reduced negative content (R_{neg} and reduced positive content (R_{pos}).

For the reduced negative content conditions (the left-hand side of Figure 1), the paper tests:

$$HR_{neg}1_0 : \Delta_{\mu_{pos}} = 0$$
$$HR_{neg}1_a : \Delta\mu_{pos} \neq 0$$

And:

$$HR_{neg}2_0 : \Delta_{\mu_{neg}} = 0$$
$$HR_{neg}2_a : \Delta_{\mu_{neg}} \neq 0$$

Then, for the reduced positive content conditions (the right-hand side of Figure 1), the paper tests:

$$HR_{pos}1_0 : \quad \Delta_{\mu_{pos}} = 0$$
$$HR_{pos}1_a : \quad \Delta\mu_{pos} \neq 0$$

And:

$$HR_{pos}2_0 : \quad \Delta_{\mu_{neg}} = 0$$
$$HR_{pos}2_a : \quad \Delta_{\mu_{neg}} \neq 0$$

Note that the theories the authors used to motivate the study imply directions for the alternative hypotheses, but nothing in the description of the analysis suggests that they used one-tailed tests. I've written these all in terms of undirected two-tailed tests to correspond to the analysis conducted in the paper. That said, given that the theories correspond to specific directions you might (arguably more accurately) have written the hypotheses in terms of directed inequalities (e.g., "$>$" or "$<$").

(b) Describing the effects

The authors' estimates suggest that reduced negative News Feed content causes an increase in the percentage of positive words and a decrease in the percentage of negative words in subsequent News Feed posts by study participants (supporting $HR_{neg}1_a$ and $HR_{neg}2_a$ respectively).

They also find that reduced positive News Feed content causes a decrease in the percentage of negative words and an increase in the percentage of positive words in subsequent News Feed posts (supporting $HR_{pos}1_a$ and $HR_{pos}2_a$)

(c) Statistical vs. practical significance

Cohen's d puts estimates of experimental effects in standardized units (much like a Z-score!) in order to help understand their size relative to the underlying distribution of the dependent variable(s). The d values for each of the effects estimated in the paper are 0.02, 0.001, 0.02, and 0.008 respectively (in the order presented in the paper, not in order of the hypotheses above). These are miniscule by the standards of most approaches to Cohen's d! However, as the authors' argue, the treatment itself is quite narrow in scope, suggesting that the presence of any treatment effect at all is an indication of the underlying phenomenon (emotional contagion). Personally, I find it difficult to attribute much substantive significance to the results because I'm not even convinced that tiny shifts in the percentage of positive/negative words used in News Feed updates accurately index meaningful emotional shifts (I might call it linguistic contagion instead?). That said, I have a hard time thinking about micro-level psychological processes and I'm probably being overly narrow/skeptical in my response. Despite these concerns and the ethical considerations that attracted so much public attention, I consider this a clever, well-executed study and I think it's quite compelling. I expect many of you will have different opinions of various kinds and I'm eager to hear about them.