

# 010 - Bootstrap Confidence Intervals

EPIB 607 - FALL 2020

Sahir Rai Bhatnagar  
Department of Epidemiology, Biostatistics, and Occupational Health  
McGill University

`sahir.bhatnagar@mcgill.ca`

slides compiled on September 30, 2020



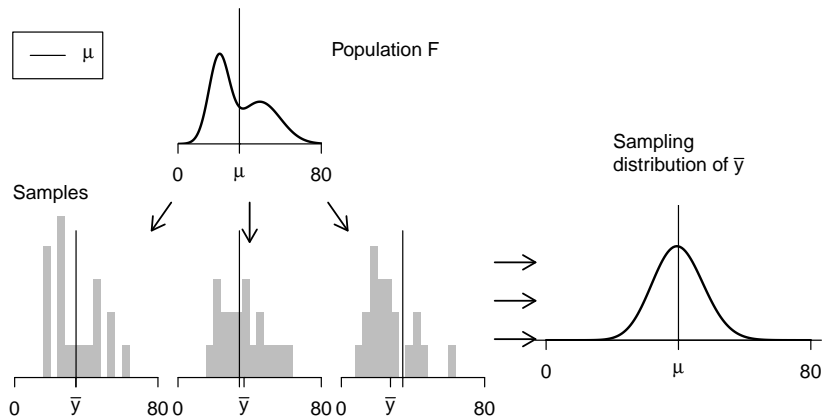


# Sampling Distribution

## Definition (Sampling Distribution)

- *The sampling distribution of a statistic is the distribution of values taken by the statistic in **all possible samples of the same size** from the same population.*
- *The standard deviation of a sampling distribution is called a **standard error***

# Sampling Distributions



**Figure:** Ideal world. Sampling distributions are obtained by drawing repeated samples from the population, computing the statistic of interest for each, and collecting (an infinite number of) those statistics as the sampling distribution

# Sampling Distribution

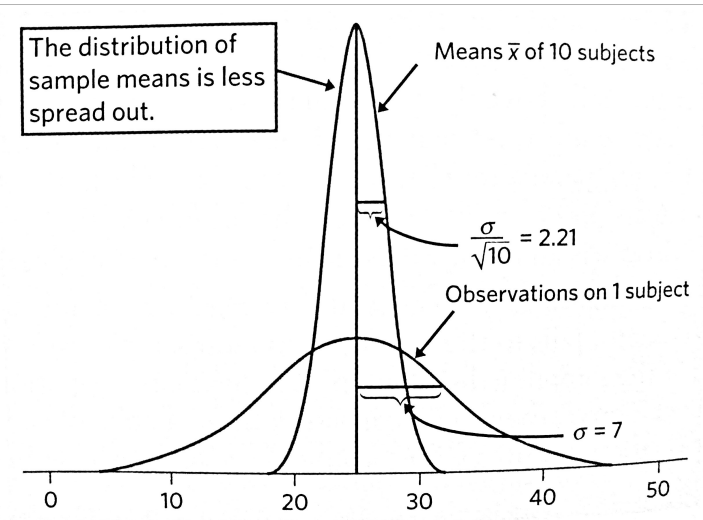


Figure: Averages are less variable than individual observations

# Traditional way to calculate CIs

How to construct a CI for the population mean?

- The **CLT** gives us that  $\bar{y} \sim \mathcal{N}(\mu, \sigma/\sqrt{n})$  is approximately true when  $n$  is large.
- We can standardize, to get  $Z = \frac{\bar{y} - \mu}{\sigma/\sqrt{n}} \sim \mathcal{N}(0, 1)$ .
- To find a CI with confidence level  $\mathcal{C} = 1 - \alpha$ , we must calculate the critical value  $z^*$  such that

$$P(-z^* < Z < z^*) = \mathcal{C} = 1 - \alpha \quad (1)$$

where  $\alpha$  is the significance level

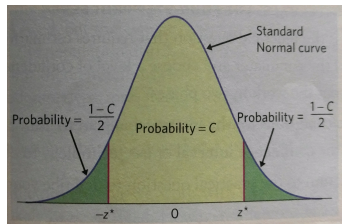
- ▶ That is, we want the value  $z^*$  that gives a *lower tail probability* of  $(1 - \mathcal{C})/2 = \alpha/2$ .
- ▶ Often this value is denoted  $z^* = z_{\alpha/2}$ ; thus we have

$$P(Z < -z_{\alpha/2}) = \alpha/2,$$

and

$$P(Z > z_{\alpha/2}) = \alpha/2.$$

# Traditional way to calculate CIs



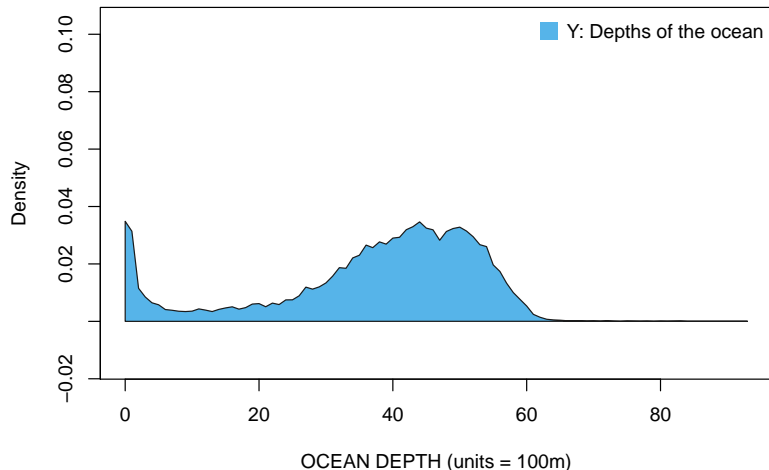
**Figure:** The critical value  $z^*$  is the number that catches central probability  $C$  under a standard normal  $\mathcal{N}(0, 1)$  curve between  $-z^*$  and  $z^*$

We can use this probability statement about the standardized version of the sample mean  $(\bar{y} - \mu)/\sigma/\sqrt{n}$ , to place bounds on where we think the true mean lies by examining the probability that  $\bar{y}$  is within  $z^* \cdot \frac{\sigma}{\sqrt{n}}$  of  $\mu$

$$\begin{aligned} C &= P\left(-z^* \leq \frac{\bar{y} - \mu}{\sigma/\sqrt{n}} \leq z^*\right) \\ &= P\left(-z^* \frac{\sigma}{\sqrt{n}} \leq \bar{y} - \mu \leq +z^* \frac{\sigma}{\sqrt{n}}\right) \\ &= P\left(-\bar{y} - z^* \frac{\sigma}{\sqrt{n}} \leq -\mu \leq -\bar{y} + z^* \frac{\sigma}{\sqrt{n}}\right) \\ &= P\left(\bar{y} + z^* \frac{\sigma}{\sqrt{n}} \geq \mu \geq \bar{y} - z^* \frac{\sigma}{\sqrt{n}}\right) \\ &= P\left(\bar{y} - z^* \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{y} + z^* \frac{\sigma}{\sqrt{n}}\right) \\ &= 1 - \alpha \end{aligned}$$

We call the interval  $\left(\bar{y} - z^* \frac{\sigma}{\sqrt{n}}, \bar{y} + z^* \frac{\sigma}{\sqrt{n}}\right)$  a **(1- $\alpha$ )100% confidence interval** for  $\mu$ .

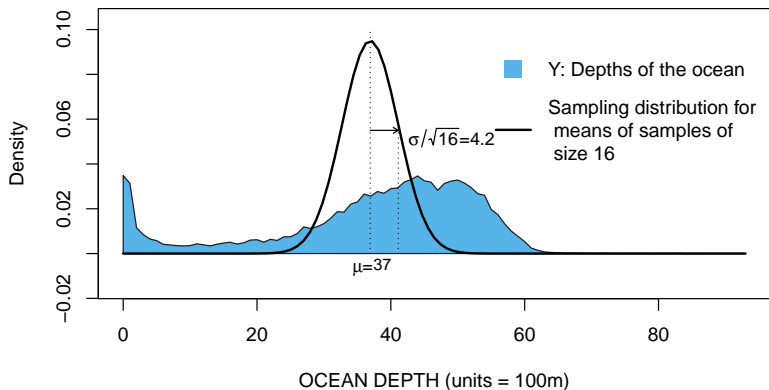
# Confidence intervals for depths of the ocean



**Figure:** The original data distribution of sampled depths of the ocean. Note that it has multiple modes and not Normal looking.



## The CLT is 'kicking in' at $n = 16$



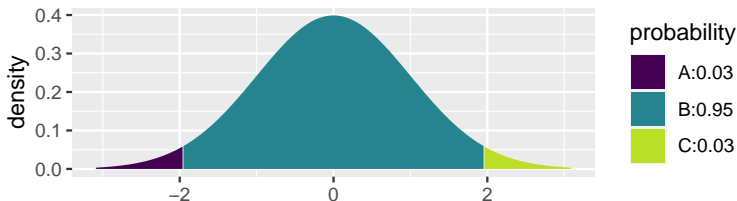
**Figure:** The sampling distribution for the mean depth of the ocean with samples of size  $n = 16$ , looks normal (centered at  $\mu = 37$  and SD equal to  $\sigma/\sqrt{16}$ )

# Since CLT has 'kicked in', we use it to construct a CI

We want to construct a  $C = 95\%$  confidence interval for the mean. Level of significance is  $\alpha = 1 - C = 0.05$

1. by the CLT  $\rightarrow \bar{y} \sim \mathcal{N}(\text{mean} = 37, \text{sd} = \sigma/\sqrt{16} = 4.2)$
2. The critical value  $z^*$  such that  $P(Z < -z^*) = P(Z > z^*) = \alpha/2 = 0.025$  is given by

```
mosaic::xqnorm(p = c(0.025, 0.975))
```



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

3. 95% CI for  $\mu$ :  $(37 - 1.96 \cdot 4.2, 37 + 1.96 \cdot 4.2) = [29, 45]$

## Alternative way of calculating CI with CLT: qnorm

- In the previous slides we used the standard normal  $\mathcal{N}(0, 1)$  to calculate the critical value  $z^*$  needed for the CI
- We were able to use the  $\mathcal{N}(0, 1)$  for two reasons:
  1. the CLT
  2. the formula used to calculate the CI is based on standardizing  $\bar{y} \rightarrow \frac{\bar{y} - \mu}{\sigma/\sqrt{n}}$
- There is an alternative, **yet equivalent**, way to calculate the CI without standardizing  $\bar{y}$ , and without using the  $\pm$  formula
- This is accomplished using qnorm
- Note: we **still need the CLT regardless** of whether we use the  $\pm$  formula or qnorm

## 68% Confidence interval using qnorm

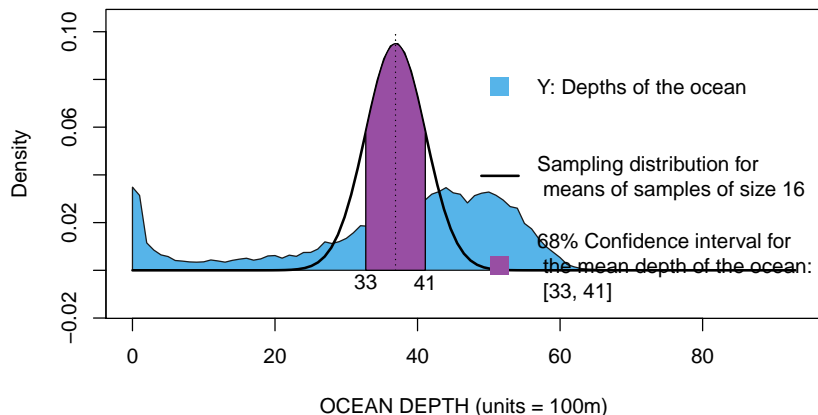
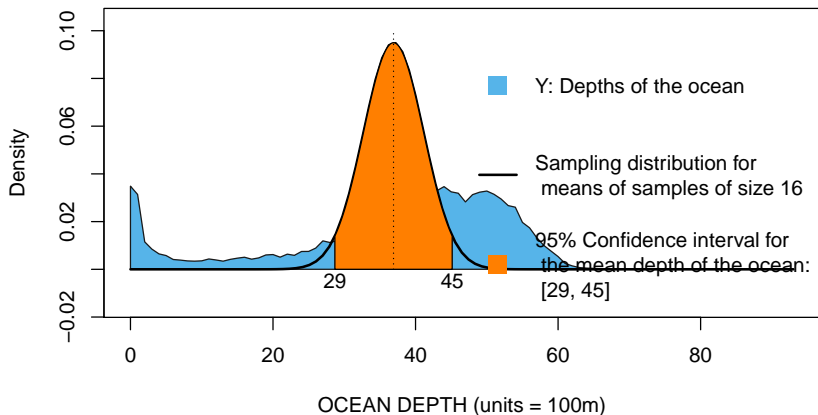


Figure: 68% Confidence interval calculated using `qnorm(p = c(0.16, 0.84), mean = 37, sd = 4.2)`

## 95% Confidence interval using qnorm



**Figure:** 95% Confidence interval calculated using `qnorm(p = c(0.025,0.975), mean = 37, sd = 4.2)`

# Motivation for the Bootstrap

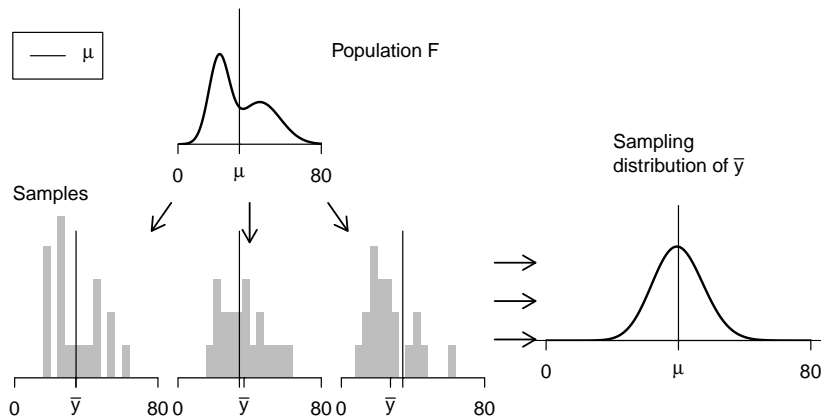
- The  $\pm$  and `qnorm` methods to calculate a CI both require the CLT

Q: What happens if the CLT hasn't 'kicked in'? Or you don't believe the CLT?

A: Bootstrap



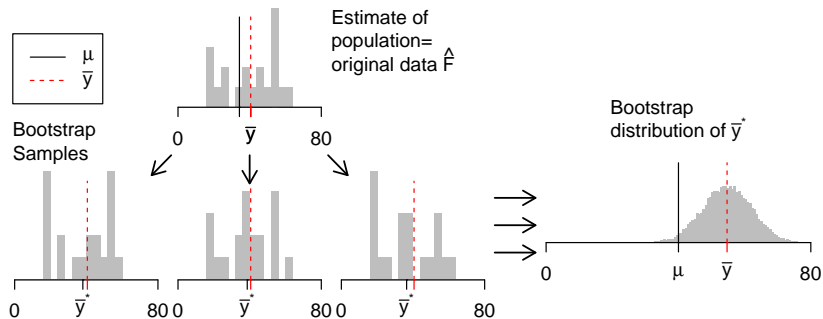
# Ideal world: known sampling distribution



**Figure:** Ideal world. Sampling distributions are obtained by drawing repeated samples from the population, computing the statistic of interest for each, and collecting (an infinite number of) those statistics as the sampling distribution



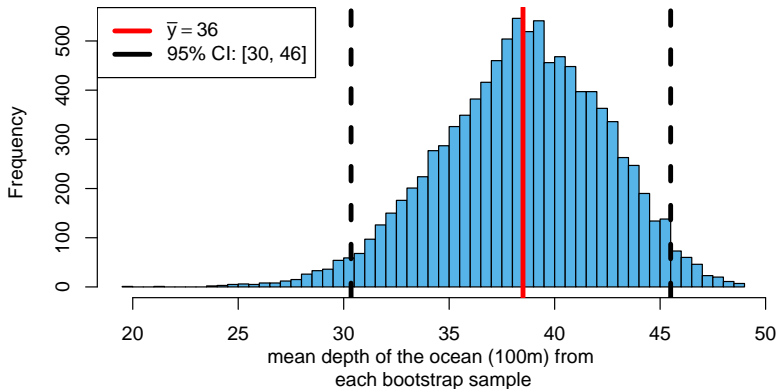
# Reality: use the bootstrap distribution instead



**Figure:** Bootstrap world. The bootstrap distribution is obtained by drawing repeated samples from an estimate of the population, computing the statistic of interest for each, and collecting those statistics. The distribution is centered at the observed statistic ( $\bar{y}$ ), not the parameter ( $\mu$ ).

# Main idea: simulate your own sampling distribution

```
R <- replicate(B, {  
  dplyr::sample_n(depths.n.20, size = N, replace = TRUE) %>%  
  dplyr::summarize(r = mean(alt)) %>%  
  dplyr::pull(r)  
})  
CI_95 <- quantile(R, probs = c(0.025, 0.975))
```



# Bootstrap code 1

```
# function for sampling ocean depths
source("https://raw.githubusercontent.com/sahirbhatnagar/EPIB607/master/labs/
003-ocean-depths/automate_water_task.R")

# from the in-class exercise
index.n.20 <- c(2106,2107,2108,2109,2110,2111,2112,
2113,2114,2115,2116,2117,2118,2119,
2120,2121,2122,2123,2124,2125)

# get depths of ocean sample n=20
depths.n.20 <- automate_water_task(index = index.n.20,
student_id = 260194225, type = "depth")

# change to 100m units
depths.n.20$alt = round(depths.n.20$alt/100,0)

B <- 10000; N <- nrow(depths.n.20)
R <- replicate(B, {
dplyr::sample_n(depths.n.20, size = N, replace = TRUE) %>%
dplyr::summarize(r = mean(alt)) %>%
dplyr::pull(r)
})

CI_95 <- quantile(R, probs = c(0.025, 0.975))
```

# Bootstrap code 2

```
# plot sampling distribution
hist(R, breaks = 50, col = "#56B4E9",
     main="",
     xlab = "mean depth of the ocean (100m) from each bootstrap sample")

# draw red line at the sample mean
abline(v = mean_depth, lty = 1, col = "red", lwd = 4)

# draw black dotted lines at 95% CI
abline(v = CI_95[1], lty = 2, col = "black", lwd = 4)
abline(v = CI_95[2], lty = 2, col = "black", lwd = 4)

# include legend
library(latex2exp)
legend("topleft",
      legend = c(TeX("$\\bar{y}$ = 36$"),
                 sprintf("95% CI: [%.f, %.f]", CI_95[1], CI_95[2])),
      lty = c(1, 2),
      col = c("red", "black"), lwd = 4)
```

# Session Info

```
R version 4.0.2 (2020-06-22)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Pop!_OS 20.04 LTS

Matrix products: default
BLAS: /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3

attached base packages:
[1] tools      stats      graphics  grDevices  utils      datasets  methods
[8] base

other attached packages:
[1] latex2exp_0.4.0  NCStats_0.4.7  FSA_0.8.30    forcats_0.5.0
[5] stringr_1.4.0    dplyr_1.0.2    purrr_0.3.4    readr_1.3.1
[9] tidyr_1.1.2      tibble_3.0.3   ggplot2_3.3.2  tidyverse_1.3.0
[13] knitr_1.29

loaded via a namespace (and not attached):
[1] fs_1.5.0          lubridate_1.7.9   RColorBrewer_1.1-2 httr_1.4.2
[5] backports_1.1.9   R6_2.4.1          DBI_1.1.0        colorspace_1.4-1
[9] withr_2.2.0       tidyrselect_1.1.0 gridExtra_2.3     leaflet_2.0.3
[13] curl_4.3          compiler_4.0.2    cli_2.0.2        rvest_0.3.6
[17] xml2_1.3.2        gg dendro_0.1.22  labeling_0.3      mosaicCore_0.8.0
[21] scales_1.1.1      digest_0.6.25     ggformula_0.9.4   foreign_0.8-79
[25] rio_0.5.16        pkgconfig_2.0.3   htmltools_0.5.0   dbplyr_1.4.4
[29] highr_0.8         htmlwidgets_1.5.1 rlang_0.4.7       readxl_1.3.1
[33] rstudioapi_0.11   farver_2.0.3      generics_0.0.2    jsonlite_1.7.1
[37] crosstalk_1.1.0.1 zip_2.1.1         car_3.0-9         magrittr_1.5
[41] mosaicData_0.20.1 Matrix_1.2-18     Rcpp_1.0.5        munsell_0.5.0
[45] fansi_0.4.1       abind_1.4-5       lifecycle_0.2.0   stringi_1.5.3
[49] carData_3.0-4     MASS_7.3-53       plyr_1.8.6        ggstance_0.3.4
[53] grid_4.0.2        blob_1.2.1        ggrepel_0.8.2     crayon_1.3.4
[57] lattice_0.20-41   haven_2.3.1       splines_4.0.2     hms_0.5.3
[61] pillar_1.4.6      reprex_0.3.0      glue_1.4.2        evaluate_0.14
[65] data.table_1.13.0 modelr_0.1.8       vctrs_0.3.4       tweenr_1.0.1
[69] cellranger_1.1.0 gtable_0.3.0      polyclip_1.10-0   assertthat_0.2.1
[73] TeachingDemos_2.12 xfun_0.17         ggforce_0.3.2     openxlsx_4.1.5
[77] broom_0.7.0       viridisLite_0.3.0 mosaic_1.7.0      ellipsis_0.3.1
```