Bootstrap (Recap)

- $X_1, X_2, \ldots, X_n \sim$ iid F (CDF)
- θ = parameter of interest — feature of F
- $\hat{F}$ = estimated CDF (parametric or nonparametric)
- $\hat{\theta}$ = estimator of θ — same feature of $\hat{F}$
- Bootstrap resample: $X_1^*, X_2^*, \ldots, X_n^* \sim$ iid $\hat{F}$

-

sample	population	Estimator
original	F	$\hat{\theta}$
Bootstrap	$\hat{F}$	$\hat{\theta}^*$

- Generate a large # (b) of resamples:

Resample #	Resample	Estimator
1	$X_{11}^*, X_{12}^*, \ldots, X_{1n}^*$	$\hat{\theta}_1^*$
2	$X_{21}^*, X_{22}^*, \ldots, X_{2n}^*$	$\hat{\theta}_2^*$
$\vdots$	$\vdots$	$\vdots$
b	$X_{b1}^*, X_{b2}^*, \ldots, X_{bn}^*$	$\hat{\theta}_b$

$\left.\begin{matrix}\hat{\theta}_1^* \\ \hat{\theta}_2^* \\ \vdots \\ \hat{\theta}_b\end{matrix}\right\}$ bootstrap dist. of $\hat{\theta}$

- If n is small; take b = # possible resamples.
- To estimate a feature η of sampling dist. of $\hat{\theta}$, use η^* = same feature estimated from the bootstrap draws of $\hat{\theta}^*$
- Essentially: "Replace θ by $\hat{\theta}$, and $\hat{\theta}$ by $\hat{\theta}^*$"
- If η = α-th quantile of $\hat{\theta} - \theta$; then $\hat{\eta}^* = \alpha$-th sample percentile of $\hat{\theta}_1^* - \hat{\theta}, \ldots, \hat{\theta}_b^* - \hat{\theta}$
$= \hat{\theta}^*_{((b+1)\alpha)} - \hat{\theta}$.

Four Bootstrap CIs for θ

1. Normal approximation CI: Use z critical point but correct $\hat{\theta}$ for bias and use $\hat{V}^*$ to estimate V.

- Estimated bias of $\hat{\theta}$ $(\hat{B}^*) = \frac{1}{b} \sum_{k=1}^{b} \hat{\theta}^*_k - \hat{\theta}$
- CI: $\left[\hat{\theta} - \hat{B}^* - z_{1-\alpha/2}\widehat{SE}^*,\ \hat{\theta} - \hat{B}^* - z_{\alpha/2}\widehat{SE}^*\right]$.

2. Studentized bootstrap CI: Use bootstrap critical point of T instead of z critical point.

$T = \frac{\hat{\theta} - \theta}{\widehat{SE}(\hat{\theta})}$.

- Get $T_1^* = (\hat{\theta}_1^* - \hat{\theta})/\widehat{SE}_1^*, \ldots, T_b^* = (\hat{\theta}_b^* - \hat{\theta})/\widehat{SE}_b^*$
- Estimated α-th percentile of T = $T^*_{((b+1)\alpha)}$
- CI: $\left[\hat{\theta} - t^*_{\left((b+1)(1-\alpha/2)\right)}\widehat{SE},\ \hat{\theta} - t^*_{\left((b+1)(\alpha/2)\right)}\widehat{SE}\right]$.

3. Basic bootstrap CI: Based on percentiles of $\hat{\theta} - \theta$ rather than $(\hat{\theta} - \theta)/\widehat{SE}$. Use bootstrap to estimate them. Notice

$$1 - \alpha = P(a_{\alpha/2} \le \hat{\theta} - \theta \le a_{1-\alpha/2})$$

$$= P[a_{\alpha/2} - \hat{\theta} \le -\theta \le a_{1-\alpha/2} - \hat{\theta}]$$

$$= P[\cancel{\hat{\theta} - a_{\alpha/2}} \le \hat{\theta} - a_{1-\alpha/2} \le \theta \le \hat{\theta} - a_{\alpha/2}]$$

α-th percentile of $\hat{\theta} - \theta$

- Estimated $a_\alpha = \hat{\theta}^*_{((b+1)\alpha)} - \hat{\theta}$

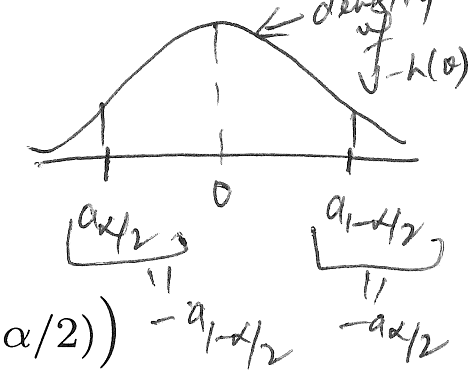
$\hat{\theta}_\alpha - \{\hat{\theta}^*_{((b+1)(1-\alpha/2))} - \hat{\theta}\}$

- CI: $\left[2\hat{\theta} - \hat{\theta}^*_{\left((b+1)(1-\alpha/2)\right)},\ 2\hat{\theta} - \hat{\theta}^*_{\left((b+1)(\alpha/2)\right)}\right]$.
- Doesn't require $\widehat{SE}$.

4. Percentile bootstrap CI: Works as in basic bootstrap → known fn. but uses "magic." Suppose there exists a transformation h so that the distribution of $h(\hat{\theta}) - h(\theta)$ is symmetric about zero. [typically ↑ fn.] Let $U = h(\hat{\theta})$. As before, we can write

⇓ h^{-1} exists.

$$
\begin{aligned}
1 - \alpha &= P\left(a_{\alpha/2} \le U - h(\theta) \le a_{1-\alpha/2}\right) \\
&\overset{\text{symmetry}}{=} P\left(-a_{1-\alpha/2} \le U - h(\theta) \le -a_{\alpha/2}\right) \\
&= P\underbrace{\left(U + \underline{a_{\alpha/2}} \le h(\theta) \le U + a_{1-\alpha/2}\right)}_{\text{CI for } h(\theta).}
\end{aligned}
$$



- Estimated $\underline{a_\alpha} =$ $\hat{U}^*_{((b+1)\alpha)} - U$
- $U + a_{\alpha/2} \approx U + \left\{U^*_{((b+1)(\alpha/2))} - U\right\} = U*_{((b+1)(\alpha/2))}$
- Similarly, $U + a_{1-\alpha/2} = U^*_{((b+1)(1-\alpha/2))}$

Therefore,

$$
\begin{aligned}
1 - \alpha &\approx P\left(U^*_{((b+1)(\alpha/2))} \le h(\theta) \le U^*_{((b+1)(1-\alpha/2))}\right) \\
&= P\left[h^{-1}\left\{U^*_{((b+1)\alpha/2)}\right\} \le h^{-1}h(\theta) \le h^{-1}\left\{U^*_{((b+1)(1-\alpha/2))}\right\}\right] \\
&= P\left[\hat{\theta}^*_{((b+1)\alpha/2)} \le \theta \le \hat{\theta}^*_{((b+1)(1-\alpha/2))}\right].
\end{aligned}
$$

- CI: $\left[\hat{\theta}^*_{\left((b+1)(\alpha/2)\right)}, \hat{\theta}^*_{\left((b+1)(1-\alpha/2)\right)}\right]$.
- Magic: No need to know 'h'.

Q. Which method to use?

Research shows that studentized bootstrap is the best choice, but it requires $\widehat{SE}$. However, if $\widehat{SE}$ is not available, then percentile bootstrap is often the next best choice. More accurate versions of this method are available.

Example: Recall the CPU time data from Example 8.12 on page 217. We had seen that a gamma distribution fit well to these data. Suppose we would like to perform inference on median cpu time.

R code:

```
# use install.packages("boot") to first install
# the package and then load it

library(boot)

# read the cpu data (we have seen these before)

> (cpu <- scan(file="cputime.txt"))
Read 30 items
 [1]  70  36  43  69  82  48  34  62  35  15  59 139
 46  37  42  30  55  56
```

```
[19]  36  82  38  89  54  25  35  24  22   9  56  19
>

# Parameter of interest: Median

############################
# Nonparametric Bootstrap #
############################

median.npar <- function(x, indices) {
  result <- median(x[indices])
  return(result)
  }

> (median.npar.boot <- boot(cpu, median.npar, R=999,
sim="ordinary", stype="i"))

ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = cpu, statistic = median.npar, R = 999,
sim = "ordinary",  stype = "i")


Bootstrap Statistics :
    original    bias    std. error
t1*     42.5 0.6721722    5.876943
>

# Let's verify the calculations

# See what's else is stored in median.npar.boot

> names(median.npar.boot)
 [1] "t0"          "t"          "R"            "data"
```

estimated using bootstrap.

sample median

the bootstrapped medians

```
 "seed"       "statistic"
 [7] "sim"        "call"       "stype"      "strata"
  "weights"
>

> median(cpu)
[1] 42.5
>
> median.npar.boot$t0
[1] 42.5
>
> mean(median.npar.boot$t)-median.npar.boot$t0
[1] 0.6721722
>
> sd(median.npar.boot$t)
[1] 5.876943
Warning message:
sd(<matrix>) is deprecated.
```

bias

se

reported by "boot"

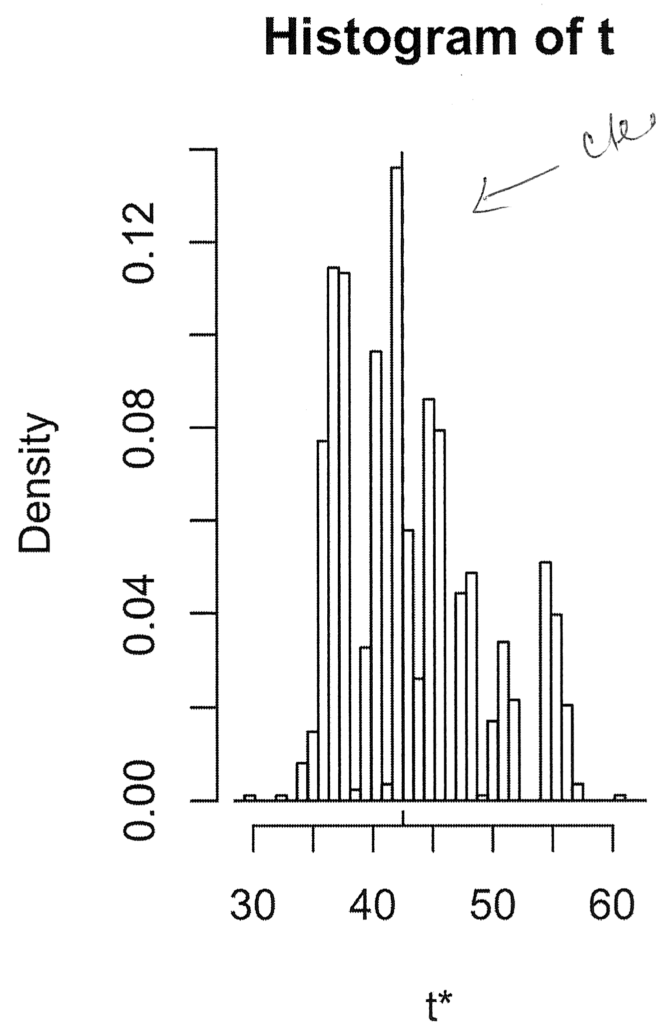
No worries!

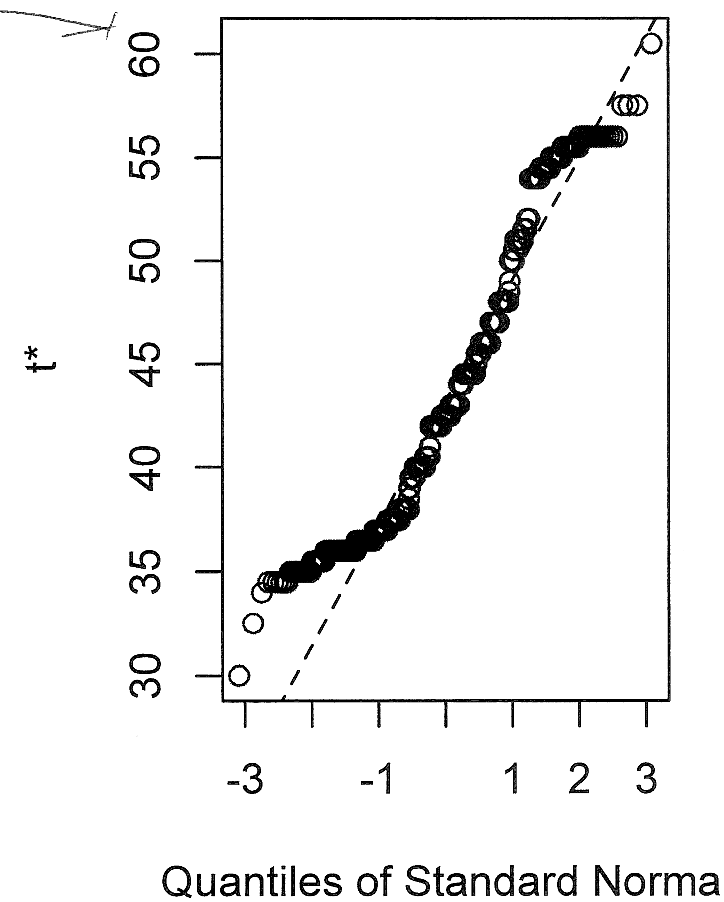
```
 Use apply(*, 2, sd) instead.
>

# See the bootstrap distribution of median estimate

plot(median.npar.boot)
```

clearly not normal ⇒ the usual z-interval won't be accurate





```
# Get the 95% confidence interval for median

> boot.ci(median.npar.boot)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = median.npar.boot)

Intervals :
Level      Normal              Basic
95%   (30.31, 53.35 )   (29.50, 49.50 )

Level     Percentile            BCa
95%   (35.5, 55.5 )   (35.0, 55.5 )
Calculations and Intervals on Original Scale
Warning message:
```

"not from this course"

```
In boot.ci(median.npar.boot) :
  bootstrap variances needed for studentized intervals
>

# Let's verify

# Normal approximation method

> c(42.5 - 0.6721722 - qnorm(0.975) * 5.876943,
    42.5 - 0.6721722 - qnorm(0.025) * 5.876943)
[1] 30.30923 53.34642
>

# Percentile bootstrap method

> sort(median.npar.boot$t)[c(25, 975)]
[1] 35.5 55.5
>
```

Everything matches with our formulas.

```
# Basic bootstrap method

> c(2*42.5-55.5, 2*42.5-35.5)
[1] 29.5 49.5
>


############################
# Parametric Bootstrap #
############################

# Saw earlier that a Gamma distribution fit well
# to these data
# > ml.est$par
# [1] 3.63149628 0.07529459
# >
# first element = shape parameter mle,
```

"boot" package is not a good choice.

```
# second element = rate parameter mle
```

$\hat{F} = \text{Gamma}(\quad , \quad)$.

```
median.par <- function(x) { return(median (x))}

resamp.par <- function(x, mle) {
  xsim <- rgamma(length(x), mle$shape, mle$rate)
  return(xsim)
  }
```

$\widehat{\text{median}}^* =$ MLE of median of Gamma dist. $(\widehat{\text{shape}}^*, \widehat{\text{rate}}^*)$.

$= \text{qgamma}(0.50, \quad , \quad)$.

```
> (median.par.boot <- boot(cpu, median.par, R=999,
sim="parametric", ran.gen=resamp.par,
mle=list(shape=3.63149628, rate=0.07529459)))
```

PARAMETRIC BOOTSTRAP

```
Call:
boot(data = cpu, statistic = median.par, R = 999, sim = "pa
    ran.gen = resamp.par, mle = list(shape = 3.63149628, ra


Bootstrap Statistics :
    original    bias    std. error
t1*     42.5 1.696431    5.220725
>


# boot uses sample median as estimator rather than its
# MLE which is the median of the fitted gamma distribution

> qgamma(0.5, shape = 3.63149628, rate = 0.07529459)
[1] 43.88304
>
```

X

↓ MLE of median based on original sample

```
> boot.ci(boot.ci(median.par.boot))
Error in empinf(boot.out, index = index, t = t.o, ...) :
  influence values cannot be found from a parametric boots
In addition: Warning message:
In boot.ci(median.par.boot) :
  bootstrap variances needed for studentized intervals
>
```

- boot package does not seem to work well for parametric bootstrap. But that's okay — this will be your ~~Mini Project 4.~~ extra credit assignment.