The Bootstrap[1]

STA431 Spring 2023

[1] See last slide for copyright information.

Overview

Sampling distributions

- Let $\mathbf{x} = (X_1, \ldots, X_n)$ be a random sample from some distribution F.
- $t = t(\mathbf{x})$ is a statistic (could be a vector of statistics).
- Need to know about the distribution of t.
- Sometimes it's not easy, even asymptotically.

Sampling distribution of t: The elementary version

For example $t = \overline{X}$

- Sample repeatedly from this population (pretend).
- For each sample, calculate t.
- Make a relative frequency histogram of the t values you observe.
- As the number of samples becomes very large, the histogram approximates the distribution of t.

Bootstrap?

Pull yourself up by your bootstraps

This photograph was taken by Tarquin. It is licensed under a Creative Commons Attribution - ShareAlike 3.0 Unported License. For more information, see the entry at the wikimedia site.

The (statistical) Bootstrap

Bradley Efron, 1979

- Select a random sample from the population.
- If the sample size is large, the sample is similar to the population.
- Sample repeatedly *from the sample.* This is called *resampling*.
- Sample from the sample? Think of putting the sample data values in a jar . . .
- Calculate the statistic for every bootstrap sample.
- A histogram of the resulting values approximates the shape of the sampling distribution of the statistic.

Notation

- Let $\mathbf{x} = (X_1, \ldots, X_n)$ be a random sample from some distribution F.
- $t = t(\mathbf{x})$ is a statistic (could be a vector of statistics).
- Form a "bootstrap sample" $\mathbf{x}^*$ by sampling n values from $\mathbf{x}$ *with replacement*.
- Repeat this process B times, obtaining $\mathbf{x}_1^*, \ldots, \mathbf{x}_B^*$.
- Calculate the statistic for each bootstrap sample, obtaining $t_1^*, \ldots, t_B^*$.
- Relative frequencies of $t_1^*, \ldots, t_B^*$ approximate the sampling distribution of t.

Why does it work?

Empirical distribution function

$$\widehat{F}(x) = \frac{1}{n}\sum_{i=1}^{n} I\{X_i \leq x\} \xrightarrow{p} E(I\{X_i \leq x\}) = F(x)$$

- Resampling from $\mathbf{x}$ with replacement is the same as simulating a random variable whose distribution is the empirical distribution function $\widehat{F}(x)$.
- Suppose the distribution function of t is a nice smooth function of F.
- Then as $n \to \infty$ and $B \to \infty$, bootstrap sample moments and quantiles of $t_1^*, \ldots, t_B^*$ converge to the corresponding moments and quantiles of the unknown distribution of t.
- If the distribution of $\mathbf{x}$ is discrete and supported on a finite number of points, the technical issues are minor.

Main Application for This Course

Skipping quantile bootstrap confidence intervals and many other interesting things

- $t = \widehat{\boldsymbol{\theta}}_n$.
- Even when the data are non-normal and the model is wrong, $\widehat{\boldsymbol{\theta}}_n$ is asymptotically normal and converges to a definite target, provided the MLE is unique.
- For the models that appear in this class,
- If the model is correct (except for the distribution) and the parameters are identifiable, $\widehat{\boldsymbol{\theta}}_n$ is consistent as well as asymptotically normal.
- The only problem is that the variances and covariances in $\mathbf{V}_n = \frac{1}{n}\mathcal{I}(\boldsymbol{\theta})$ may be wrong.
- Need a different asymptotic covariance matrix (sometimes).

Bootstrap the covariance matrix of $\widehat{\boldsymbol{\theta}}_n$

- Asymptotic distribution is multivariate normal
- Centered on the right thing.
- The only other thing we need to know about the distribution of $\widehat{\boldsymbol{\theta}}_n$ is its covariance matrix.

Procedure

- The data ‘jar” contains not balls with single numbers, but strings of beads with a vector of observed values $\mathbf{d}_i$ written on them. Data values for a case stay together.
- Select n strings of beads with replacement, obtaining $\mathbf{x}_1^*$. .
- Do this B times. Now you have $\mathbf{x}_1^*, \ldots, \mathbf{x}_B^*$.
- Calculate $\widehat{\boldsymbol{\theta}}_1^*, \ldots \widehat{\boldsymbol{\theta}}_B^*$.
- You have a lot of information about the multivariate distribution of $\widehat{\boldsymbol{\theta}}_n$, but all you care about is the covariance matrix.
- If there are m parameters, you have a $B \times m$ matrix of numbers, with one column for each parameter in the model.
- Calculate the sample covariance matrix for the data (using `var`).
- This is the new $\widehat{\mathbf{V}}_n$.
- Use it for Wald tests and z-tests.
- All this applies to MOM as well as MLE.

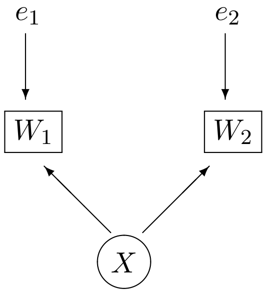
Sometimes it's Unnecessary

- Linear structural equation models have a lot of robustness to the multivariate normal assumption.
- When it fails, it's usually for data with "excess kurtosis" (heavy tails).
- And even then, not necessarily for all parameters.
- Trouble arises when the variance of the sample variance is involved.

$$Var\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\overline{x}_n)^2\right)$$

  Fourth moments of the normal distribution will be too small, leading to an under-estimate.
- For the double measurement design, standard errors of the regression coefficients are robust to normality.

Example: Double measurement


$$\begin{tikzcd} e_1 \arrow[d] & & e_2 \arrow[d] \\ \boxed{W_1} & & \boxed{W_2} \\ & X \arrow[ul] \arrow[ur] & \end{tikzcd}$$

$$\begin{aligned} W_1 &= X + e_1 \\ W_2 &= X + e_2, \end{aligned}$$

where $E(X) = \mu$, $Var(X) = \phi$, $E(e_1) = E(e_2) = 0$, $Var(e_1) = \omega_1$, $Var(e_2) = \omega_2$, and X, e_1 and e_2 are all independent.

Equivalent measurements?

$$\begin{tikzcd}
e_1 \arrow[d] & & e_2 \arrow[d] \\
\boxed{W_1} & & \boxed{W_2} \\
& X \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

If $\omega_1 = Var(e_1)$ and $\omega_2 = Var(e_2)$ are equal, W_1 and W_2 are *equivalent measurements*, and $Corr(W_1, W_2) = \frac{\phi}{\phi+\omega}$, the reliability.

Simulate from the t Distribution: Heavy-tailed

$Var(t) = \nu/(\nu-2)$, so with $\nu = 3$, $Var(t) = 3$

```
> rm(list=ls())
> # Parameter values and sample size
> phi = 7; omega1 = 3; omega2 = 3
> rel1 = round(phi/(phi+omega1),3); rel2 = round(phi/(phi+omega2),3)
> c(rel1,rel2) # Reliabilities
[1] 0.7 0.7
> n = 1500
> # Simulate from t distribution -- heavy tails
> # Var(t) = nu/(nu-2)
> set.seed(9999)
> x = sqrt(phi) * rt(n,3)/sqrt(3)
> e1 = sqrt(omega1) * rt(n,3)/sqrt(3); e2 = sqrt(omega2) * rt(n,3)/sqrt(3)
> w1 = x + e1; w2 = x + e2
> ww = cbind(w1,w2)
> vcovW = var(ww) * (n-1)/n; vcovW # Divide by n to get MLEs
          w1       w2
w1 10.120663 6.727376
w2  6.727376 9.347715
```

Normal Theory Fit with `lavaan`

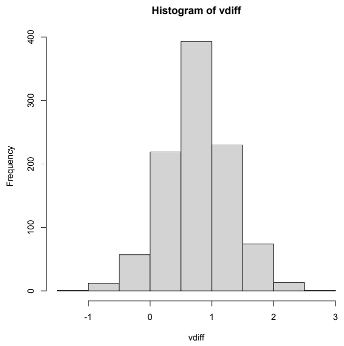
```
> # install.packages("lavaan", dependencies = TRUE) # Only need to do this once
> library(lavaan)
This is lavaan 0.6-11
lavaan is FREE software! Please report any bugs.
> # Normal theory with lavaan
> mod =   "x =~ 1.0*w1 + 1.0*w2
+          x ~~ phi*x; w1 ~~ omega1*w1; w2 ~~ omega2*w2
+          vardiff := omega1-omega2
+          "
> fit = lavaan(mod, data=ww)
> # summary(fit)
> parameterEstimates(fit)
      lhs op           rhs   label   est    se      z pvalue ci.lower ci.upper
1       x =~            w1         1.000 0.000     NA     NA    1.000    1.000
2       x =~            w2         1.000 0.000     NA     NA    1.000    1.000
3       x ~~             x     phi 6.727 0.305 22.031  0.000    6.129    7.326
4      w1 ~~            w1  omega1 3.393 0.220 15.448  0.000    2.963    3.824
5      w2 ~~            w2  omega2 2.620 0.205 12.778  0.000    2.218    3.022
6 vardiff := omega1-omega2 vardiff 0.773 0.364  2.124  0.034    0.060    1.486
> thetahat = coef(fit); thetahat
   phi omega1 omega2
 6.727  3.393  2.620
```

Bootstrap

```
> # Bootstrap the "hard" way
> # n = dim(ww)[1] is not needed
> jar = 1:n; B = 1000
> tstar = matrix(NA,B,3) # Rows will hold theta-hat values
> colnames(tstar) = names(coef(fit))
> for(j in 1:B)
+     {
+     rowz = sample(jar,size=n,replace=TRUE)
+     xstar = ww[rowz,]
+     fitstar = lavaan(mod, data=xstar)
+     tstar[j,] = coef(fitstar)
+     } # Next bootstrap sample
> head(tstar)
          phi   omega1   omega2
[1,] 6.969279 4.360700 2.182922
[2,] 6.324895 4.075226 2.259924
[3,] 6.607809 3.034017 2.047602
[4,] 6.931564 3.314822 3.254835
[5,] 6.157233 3.992400 2.434781
[6,] 8.465813 3.019230 2.719412
```

Sampling Distribution of $\widehat{\omega}_1 - \widehat{\omega}_2$

```
> vdiff = tstar[,2] - tstar[,3] # Vector of omega1hat - omega2hat values
> hist(vdiff)
```



```
> shapiro.test(vdiff) # Test of normality
Shapiro-Wilk normality test

data:  vdiff
W = 0.99873, p-value = 0.7097
```

Standard error of $\widehat{\omega}_1 - \widehat{\omega}_2$

```
> var(vdiff)
[1] 0.2889961
> bootse = sqrt(var(vdiff))
> bootse # Compare normal theory estimate of 0.364
[1] 0.5375836
> z = (thetahat[2]-thetahat[3])/bootse; z # Compare z = 2.124
  omega1
1.437819
> # Now bootstrap with lavaan: The easy way
> fitB = lavaan(mod, data=ww, se = "bootstrap")
> parameterEstimates(fitB)
      lhs op           rhs   label   est    se     z pvalue ci.lower ci.upper
1       x =~            w1           1.000 0.000    NA     NA    1.000    1.000
2       x =~            w2           1.000 0.000    NA     NA    1.000    1.000
3       x ~~             x     phi 6.727 0.922 7.295  0.000    5.255    8.734
4      w1 ~~            w1  omega1 3.393 0.386 8.781  0.000    2.685    4.213
5      w2 ~~            w2  omega2 2.620 0.353 7.419  0.000    1.996    3.390
6 vardiff := omega1-omega2 vardiff 0.773 0.525 1.473  0.141   -0.192    1.833
```

Advantages and Disadvantages

Of bootstrapping the normal MLEs

Advantages

- No assumptions about the distribution of the data.
- Works for *any* linear structural equation model provided the observed data have finite fourth moments.
- It's easy.

Disadvantages

- It might take a minute or two.
- The answer is slightly different every time.
- You need the raw data.

$$L(\boldsymbol{\mu},\boldsymbol{\Sigma}) = |\boldsymbol{\Sigma}|^{-n/2}(2\pi)^{-np/2}\exp -\frac{n}{2}\left\{tr(\widehat{\boldsymbol{\Sigma}}\boldsymbol{\Sigma}^{-1}) + (\overline{\mathbf{d}}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\overline{\mathbf{d}}-\boldsymbol{\mu})\right\}$$

Copyright Information

This slide show was prepared by Jerry Brunner, Department of Statistics, University of Toronto. It is licensed under a Creative Commons Attribution - ShareAlike 3.0 Unported License. Use any part of it as you like and share the result freely. The LaTeX source code is available from the course website:

http://www.utstat.toronto.edu/brunner/oldclass/431s23