

Module 7: Simulations

Siyue Yang

06/03/2022

Outline

In this module, we will review

- Simulation study
- Rationale for simulations
- Simulation procedure for estimation and hypothesis testing through simple examples
- Tips for running simulations

Simulation study

- Simulation: A numerical techniques for conducting experiments on the computer
- Monte Carlo simulation: Computer experiment involving random sampling from probability distributions

Why simulation?

To establish/validate the properties of statistical methods

- Exact analytical derivations of properties are **rarely** possible
- Large sample approximations to properties are **often possible**, but need to evaluate their relevance to (finite) sample sizes likely to be encountered in practice

Why simulation?

To establish/validate the properties of statistical methods

- Exact analytical derivations of properties are **rarely** possible
- Large sample approximations to properties are **often possible**, but need to evaluate their relevance to (finite) sample sizes likely to be encountered in practice

Moreover, analytical results may require **assumptions** (e.g., normality)

- But what happens when these assumptions are violated?
- Analytical results, even large sample ones, may not be possible

Considerations for simulation

- Is an estimator **biased** in finite samples? Is it still **consistent** under departures from assumptions? What is its **sampling variance**?
- How does it **compare** to competing estimators on the basis of bias, precision, etc.?

Considerations for simulation

- Is an estimator **biased** in finite samples? Is it still **consistent** under departures from assumptions? What is its **sampling variance**?
- How does it **compare** to competing estimators on the basis of bias, precision, etc.?
- Does a procedure for constructing a **confidence interval** for a parameter achieve the advertised **nominal level of coverage**?
- Does a **hypothesis testing** procedure attain the advertised **level** or **size**?
- If it does, what **power** is possible against different alternatives to the null hypothesis? Do different test procedures deliver different power?

Monte Carlo simulation

- Generate S independent data sets under the conditions of interest
- Compute the numerical value of the estimator/test statistic T (data) for each data set $\Rightarrow T_1, \dots, T_S$
- If S is large enough, **summary statistics** across T_1, \dots, T_S should be good **approximations** to the true sampling properties of the estimator/test statistic under the conditions of interest

Simulations for properties of estimators

Example: Compare 3 estimators for the **mean** μ of a distribution based on i.i.d. draws Y_1, \dots, Y_n

- Sample mean $T^{(1)}$
- Sample 20% trimmed mean $T^{(2)}$
- Sample median $T^{(3)}$

Simulations for properties of estimators (cont'd)

Simulation procedure: For a particular choice of μ , n , and true underlying distribution

- Generate independent draws Y_1, \dots, Y_n from the distribution
- Compute $T^{(1)}, T^{(2)}, T^{(3)}$
- Repeat S times $T_1^{(1)}, \dots, T_S^{(1)}; \quad T_1^{(2)}, \dots, T_S^{(2)}; \quad T_1^{(3)}, \dots, T_S^{(3)}$
- Compute for $k = 1, 2, 3$

$$\widehat{\text{mean}} = S^{-1} \sum_{s=1}^S T_s^{(k)} = \bar{T}^{(k)}, \quad \widehat{\text{bias}} = \bar{T}^{(k)} - \mu$$

$$\widehat{\text{SD}} = \sqrt{(S-1)^{-1} \sum_{s=1}^S \left(T_s^{(k)} - \bar{T}^{(k)} \right)^2}$$

$$\widehat{\text{MSE}} = S^{-1} \sum_{s=1}^S \left(T_s^{(k)} - \mu \right)^2 \approx \widehat{\text{SD}}^2 + \widehat{\text{bias}}^2$$

Simulations for properties of estimators (cont'd)

Another important property we care about is the **relative efficiency** (RE).

- If the estimators are unbiased,

$$RE = \frac{\text{var} \left(T^{(1)} \right)}{\text{var} \left(T^{(2)} \right)}$$

- If the estimators are biased,

$$RE = \frac{\text{MSE} \left(T^{(1)} \right)}{\text{MSE} \left(T^{(2)} \right)}$$

In either case $RE < 1$ means estimator 1 is preferred (estimator 2 is inefficient relative to estimator 1 in this sense)

Set up parameters

```
set.seed(3)
S <- 1000
n <- 15
mu <- 1
sigma <- sqrt(5/3)

trimmean <- function(Y) mean(Y, 0.2)
```

Generate data

Note: for this very simple data generation, we can get the data in one step, no looping. In more complex statistical models, looping is often required.

```
generate.normal <- function(S, n, mu, sigma){  
  dat <- matrix(rnorm(n*S, mu, sigma), ncol=n, byrow=T)  
  out <- list(dat=dat)  
  return(out)  
}
```

```
out <- generate.normal(S, n, mu, sigma)  
out_mean <- apply(out$dat, 1, mean)  
out_trimmean <- apply(out$dat, 1, trimmean)  
out_median <- apply(out$dat, 1, median)
```

View the simulated data

```
summary.sim <- data.frame(mean = out_mean,  
                           trim = out_trimmean,  
                           median = out_median)
```

```
head(summary.sim)
```

##		mean	trim	median
## 1		0.753935	0.7131731	1.0388898
## 2		0.643902	0.4580396	0.3745711
## 3		1.555288	1.6710299	1.9394763
## 4		0.517147	0.4826527	0.4118927
## 5		1.360281	1.4620501	1.3451583
## 6		1.359185	1.3955097	1.4949135

View the estimator properties

```
simsum <- function(dat, trueval){  
  
  S <- nrow(dat)  
  
  MCmean <- apply(dat,2,mean)  
  MCbias <- MCmean-trueval  
  MCrelbias <- MCbias/trueval  
  MCstddev <- sqrt(apply(dat,2,var))  
  MCMSE <- apply((dat-trueval)^2,2,mean)  
  # MCMSE <- MCbias^2 + MCstddev^2 # alternative lazy calculation  
  MCRE <- MCMSE[1]/MCMSE  
  
  sumdat <- rbind(rep(trueval,3), S, MCmean, MCbias,  
                  MCrelbias, MCstddev, MCMSE, MCRE)  
  names <- c("true value", "# sims", "MC mean", "MC bias", "MC relative bias",  
             "MC standard deviation", "MC MSE", "MC relative efficiency")  
  ests <- c("Sample mean", "Trimmed mean", "Median")  
  
  dimnames(sumdat) <- list(names,ests)  
  round(sumdat,5)  
}
```

View the estimator properties (cont'd)

```
results <- simsum(summary.sim, mu)
results
```

##	Sample mean	Trimmed mean	Median
## true value	1.00000	1.00000	1.00000
## # sims	1000.00000	1000.00000	1000.00000
## MC mean	0.98515	0.98690	0.99173
## MC bias	-0.01485	-0.01310	-0.00827
## MC relative bias	-0.01485	-0.01310	-0.00827
## MC standard deviation	0.33088	0.34800	0.39763
## MC MSE	0.10959	0.12116	0.15802
## MC relative efficiency	1.00000	0.90456	0.69356

Performance of estimates of uncertainty

How well do estimated standard errors represent the true sampling variation?

- Compare the average of the estimated standard errors to MC standard deviation.
- For sample mean \bar{Y} , $SE(\bar{Y}) = \frac{s}{\sqrt{n}}$, $s^2 = (n-1)^{-1} \sum_{j=1}^n (Y_j - \bar{Y})^2$

```
results["MC standard deviation", "Sample mean"]
```

```
## [1] 0.33088
```

```
mean_se <- sqrt(apply(out$dat, 1, var)/n)
ave_mean_se <- mean(mean_se)
round(ave_mean_se, 3)
```

```
## [1] 0.329
```

Confidence interval

Based on the sample mean,

$$\left[\bar{Y} - t_{1-\alpha/2, n-1} \frac{s}{\sqrt{n}}, \bar{Y} + t_{1-\alpha/2, n-1} \frac{s}{\sqrt{n}} \right]$$

Does the interval achieve the nominal level of coverage $1 - \alpha$?

```
t05 <- qt(0.975, n-1)

coverage <- sum((out_mean - t05*mean_se <= mu) &
               (out_mean + t05*mean_se >= mu))/S
coverage

## [1] 0.949
```

Simulations for properties of hypothesis testing

Example: Size and power of the usual t -test for the mean

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$

Simulations for properties of hypothesis testing

Example: Size and power of the usual t -test for the mean

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$

To evaluate whether size/level of test achieves advertised α

- Approximates the true probability of rejecting H_0 when it is true
- Generate data under $H_0 : \mu = \mu_0$
- Calculate proportion of rejections of H_0 , should $\approx \alpha$

Simulations for properties of hypothesis testing

Example: Size and power of the usual t -test for the mean

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$

To evaluate whether size/level of test achieves advertised α

- Approximates the true probability of rejecting H_0 when it is true
- Generate data under $H_0 : \mu = \mu_0$
- Calculate proportion of rejections of H_0 , should $\approx \alpha$

To evaluate the power

- Approximates the true probability of rejecting H_0 when the alternative is true (power)
- Generate data under some alternative $H_1 : \mu \neq \mu_0$
- Calculate proportion of rejections of H_0

Parameters set up

```
set.seed(3)
S <- 1000
n <- 15
sigma <- sqrt(5/3)
```

Size/level of test

```
mu0 <- 1  
mu <- 1  
out <- generate.normal(S, n, mu, sigma)
```

```
samp_mean <- apply(out$dat, 1, mean)  
mean_se <- sqrt(apply(out$dat, 1, var)/n)  
ttests <- (samp_mean - mu0)/mean_se
```

```
t05 <- qt(0.975, n-1)  
sum(abs(ttests) > t05)/S
```

```
## [1] 0.051
```

Power of test

```
mu0 <- 1
mu <- 1.75
out <- generate.normal(S, n, mu, sigma)
```

```
samp_mean <- apply(out$dat, 1, mean)
mean_se <- sqrt(apply(out$dat, 1, var)/n)
ttests <- (samp_mean - mu0)/mean_se
```

```
t05 <- qt(0.975, n-1)
sum(abs(ttests) > t05)/S
```

```
## [1] 0.512
```


Tips for Running Your Own Simulation Studies

① Setting parameter values:

- First run your code under a favorable setting (make sure it works)
- Then choose parameter values that will challenge your method
- Is $S = 1000$ large enough to get a feel for the true sampling properties? How “believable” are the results?

Carefully choosing S

Estimator for θ (true value θ_0) e.g. mean of sampling distribution

$$\sqrt{\text{var}(\bar{T} - \theta_0)} = \sqrt{\text{var}(\bar{T})} = \sqrt{\text{var}\left(S \sum_{s=1}^S T_s\right)} = \frac{\text{SD}(T_s)}{\sqrt{S}} = d$$

where d is the acceptable error

$$\Rightarrow S = \frac{\{\text{SD}(T_s)\}^2}{d^2}$$

Carefully choosing S (cont'd)

Coverage probabilities, size, power e.g. for a hypothesis testing

$$Z = \# \text{ rejections} \sim \text{binomial}(S, p) \Rightarrow \sqrt{\text{var}\left(\frac{Z}{S}\right)} = \sqrt{\frac{p(1-p)}{S}}$$

- Worst case is at $p = 1/2 \Rightarrow 1/\sqrt{4S}$
- d acceptable error $\Rightarrow S = 1/(4d^2)$; e.g., $d = 0.01$ yields $S = 2500$
- For coverage, size, $p = 0.05$

Tips for Running Your Own Simulation Studies

- ① Setting parameter values:
 - First run your code under a favorable setting (make sure it works)
 - Then choose parameter values that will challenge your method
- ② Don't make B too large to start (≈ 500)

Tips for Running Your Own Simulation Studies

- ① Setting parameter values:
 - First run your code under a favorable setting (make sure it works)
 - Then choose parameter values that will challenge your method
- ② Don't make B too large to start (≈ 500)
- ③ Save all the estimates and not just the summary statistics

Save everything

- Save individual estimates in a file then analyze
- Useful when simulation takes a long time to run

```
# Save txt file.
```

```
file_name <- paste0("ssl_binary",  
                    "_lab", n,  
                    "_beta", b,  
                    "_prev", p,  
                    "_setting", 1,  
                    "_reps", n_sim,  
                    ".txt")  
  
write.table(result, file = out_file,  
            sep = "\t", row.names = FALSE)
```

```
# Save .Rdata file
```

```
save(result)
```

Tips for Running Your Own Simulation Studies

- ① Setting parameter values:
 - First run your code under a favorable setting (make sure it works)
 - Then choose parameter values that will challenge your method
- ② Don't make B too large to start (≈ 500)
- ③ Save all the estimates and not just the summary statistics
- ④ Set the seed

Set a different seed for each run and keep records

- Ensure simulation runs are independent
- Runs may be replicated if necessary

e.g.

```
data_generation <- function(S) {  
  
  for (i in c(1:S)) {  
    set.seed(1234+i)  
    X <- ...  
    Y <- ...  
  }  
  
  data.frame(X = X, Y = Y)  
}
```


Tips for Running Your Own Simulation Studies

- ① Setting parameter values:
 - First run your code under a favorable setting (make sure it works)
 - Then choose parameter values that will challenge your method
- ② Don't make B too large to start (≈ 500)
- ③ Save all the estimates and not just the summary statistics
- ④ Set the seed
- ⑤ Document the code (i.e. comments)
 - Keep track of the versions of the code you use (i.e. use GitHub)

Tips for Running Your Own Simulation Studies

- ① Setting parameter values:
 - First run your code under a favorable setting (make sure it works)
 - Then choose parameter values that will challenge your method
- ② Don't make B too large to start (≈ 500)
- ③ Save all the estimates and not just the summary statistics
- ④ Set the seed
- ⑤ Document the code (i.e. comments)
 - Keep track of the versions of the code you use (i.e. use GitHub)
- ⑥ If you use Rmarkdown, use the `cache=TRUE` preamble
 - Your code will only be knitted/run the first time or anytime after it updated. Saves time!

Resources

This tutorial is based on

- Marie Davidian's STA810A Preparation for Statistical Research handout of simulation studies in statistics [\[links\]](#).
- Harvard's Biostatistics Preparatory Course Methods [\[links\]](#).