Statistical methods

Lab 2

V. Gioia (and N.Torelli and G. Di Credico)

vincenzo.gioia@units.it

Office hour: Friday, 17.00 - 18.30

11/10/2022

Contents

Monte Carlo simulation

Simulation: Approximate a process and retrieve general results by assuming to observe the process several times.

We rely on the so called **Monte Carlo** simulation, a wide class of simulation procedures based on sampling independent and identically distributed values from a process —precisely, from the underlying presumably true probability distribution of the process— and computing some numerical outputs.

In what follows, we will consider the behavior of some sample statistics, such as the mean and the variance, through their sample distribution.

The **steps of a Monte Carlo simulation** are:

- **Generate n independent and identically distributed values from a process;**
- **Compute a summary for this sample, a statistic;**
- **Repeat the steps above R times and obtain a sample distribution for the statistic.**

Distribution of the sample mean

Suppose that $Y_1, \dots, Y_n$, is sequence of iid rv from

- Case 1: $\mathcal{N}(0, 1)$ (i.e. standard normal);
- Case 2: t_3 (i.e. t-student with 3 degrees of freedom);
- Case 3: $\mathrm{U}(0, 1)$ (i.e. standard continuous uniform).

Goal: Determine the distribution of the sample mean $\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$

We can answer using theoretical results, but let's investigate via simulation. Consider:

- Number of simulations $R = 1000$;
- Sample size $n = 30$.

```r
set.seed(1234)
R <- 1000
n <- 30

#generate the sample of size n (for the 3 distributions) R times
samples <- array(0, c(3, n, R))
for(i in 1 : R){
  samples[1, ,i] <- rnorm(n, 0, 1)
  samples[2, ,i] <- rt(n, df = 3)
  samples[3, ,i] <- runif(n, 0, 1)
}
```

Above we saved the results in a 3-dimensional array. Otherwise you can create 3 different matrix (of size $n \times R$), or according to the goal of the analysis you can compute directly some quantity of interest without saving the samples that you generate

Let's compute the sample mean via

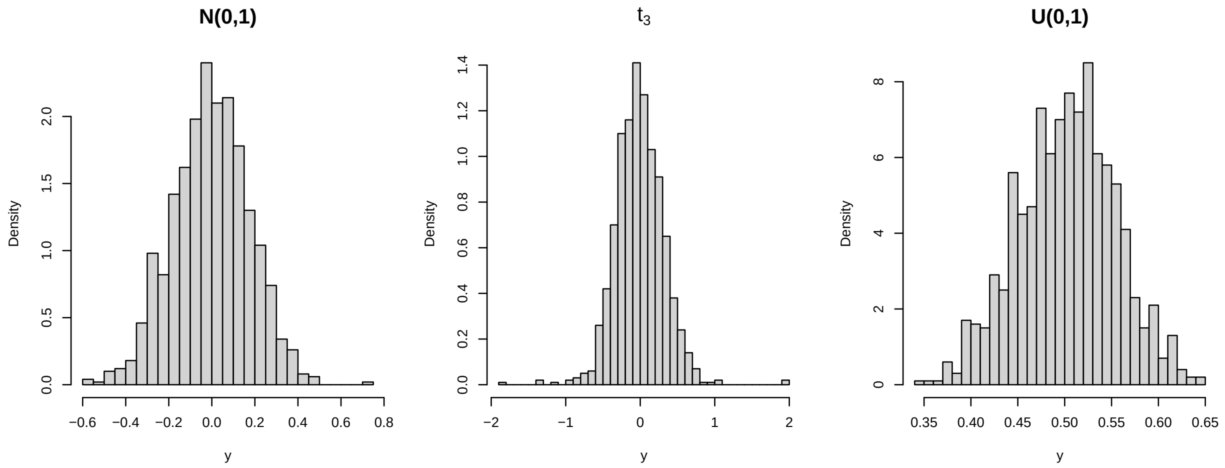
- **apply ()**
- **for loop**

```
sample_stat <- apply(samples, MARGIN = c(1, 3), FUN = mean)

# Equivalently (creating a)
sample_stat2 <- matrix(0, nrow = 3, ncol = R)
for(i in 1 : R) {
  sample_stat2[, i] <- apply(samples[, , i], 1, mean)
}
max(abs(sample_stat - sample_stat2)) # Check
```

```
## [1] 0
```

So, we can visualise via histograms the sample distribution of the sample means under the three cases

```
par(mfrow = c(1, 3))
hist(sample_stat[1,], nclass = 30, probability = TRUE,
     xlab="y", main= "N(0,1)", cex.main = 1.5)
hist(sample_stat[2,], nclass = 30, probability = TRUE,
     xlab = "y", main = expression(t[3]), cex.main = 1.5)
hist(sample_stat[3,], nclass = 30, probability = TRUE,
     xlab = "y", main = "U(0,1)", cex.main = 1.5)
```



In the Gaussian case the **sample mean** $\overline{Y}$ for iid values still follows a normal distribution, precisely $\bar{Y}_n \sim \mathcal{N}(\mu, \sigma^2/n)$.

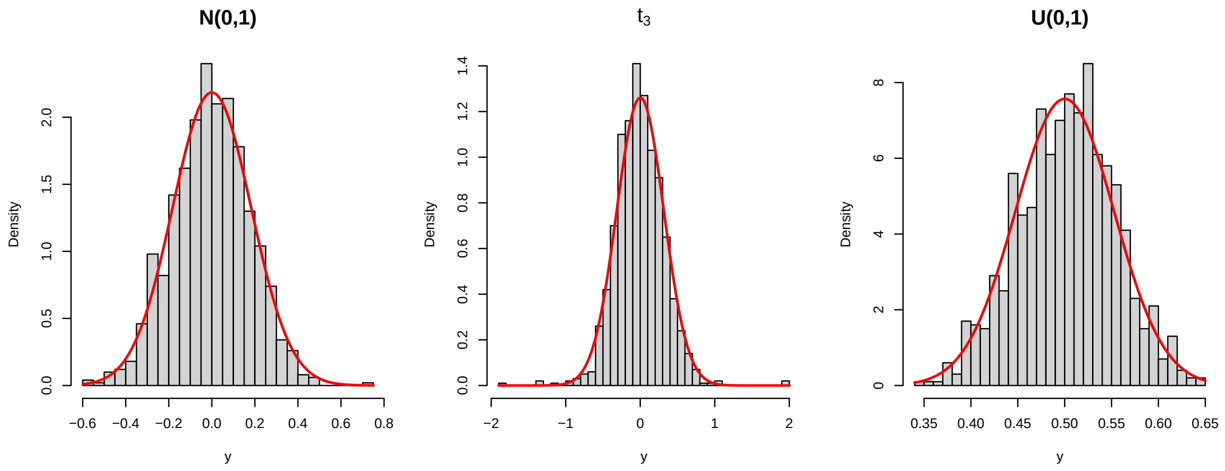
For other distributions, this result holds only asymptotically, when $n \to \infty$, due to the CLT theorem. When the sampling does not come from a Normal distribution $\overline{Y} \dot{\sim} \mathcal{N}(\mu, \sigma^2/n)$.

Time to working with R: It's your turn

According to the previous comment, we want overlap the proper Gaussian density over the histograms (the final result is reported below)

- This requires obtaining the parameters of the appropriate Gaussian distribution. Thus, for each distribution we must obtain $E[\bar{Y}_n]$ and $Var(\bar{Y}_n)$

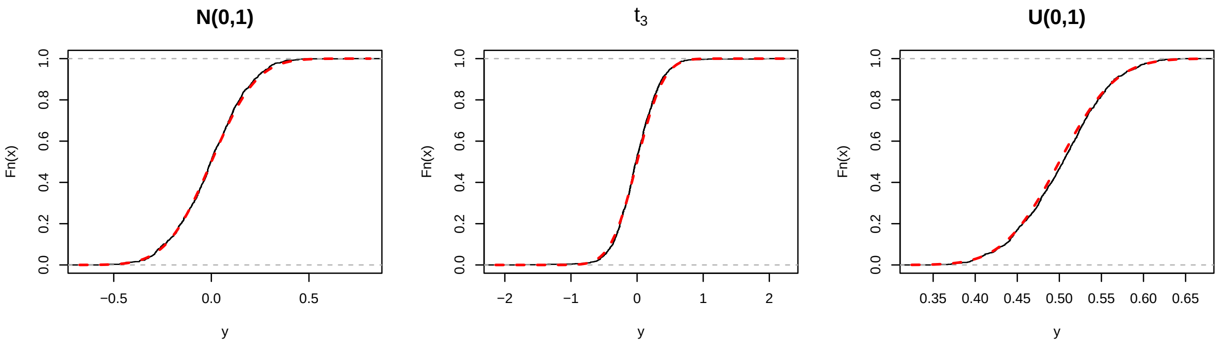
```
par (mfrow=c(1,3))

hist(sample_stat[1,], nclass = 30, probability = TRUE,
     xlab="y", main= "N(0,1)", cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lwd = 2)

hist(sample_stat[2,], nclass = 30, probability = TRUE,
     xlab = "y", main = expression(t[3]), cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lwd = 2)

hist(sample_stat[3,], nclass = 30, probability = TRUE,
     xlab = "y", main = "U(0,1)", cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lwd = 2)
```



Other graphical tools:

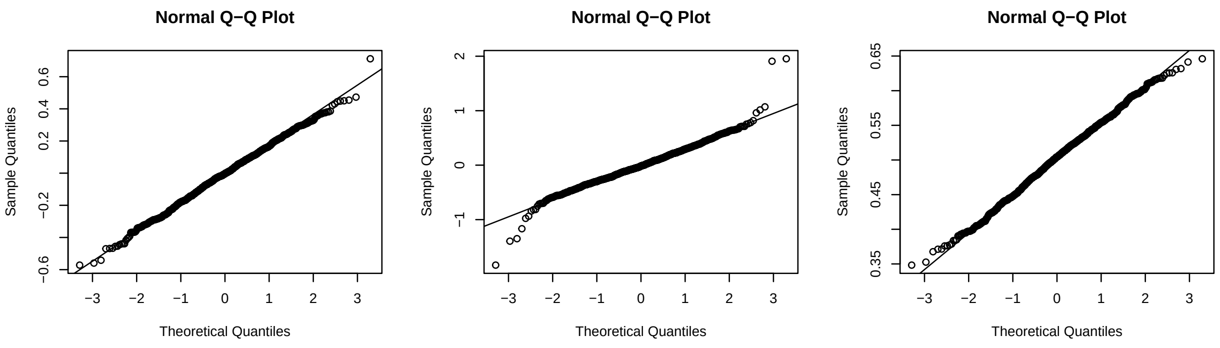
- Overlapping the empirical cumulative distribution function (ecdf) to the proper Gaussian cumulative distribution function

```
plot(ecdf(sample_stat[1,]), xlab="y", main= "N(0,1)", cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lty = 2, lwd = 2)

plot(ecdf(sample_stat[2,]), xlab="y", main= expression(t[3]), cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lty = 2, lwd = 2)

plot(ecdf(sample_stat[3,]), xlab="y", main= "U(0,1)", cex.main = 1.5)
curve(XXX, add = TRUE, col = "red", lty = 2, lwd = 2)
```



- Analyse the qq-norm plot (during theory lecture, you have seen the function qqPlot())

```
qqnorm(sample_stat[1,])
abline(XXX, XXX)

qqnorm(sample_stat[2,])
abline(XXX, XXX)

qqnorm(sample_stat[3,])
abline(XXX, XXX)
```



Distribution of the sample variance under the Gaussian case

We know that $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$ is an unbiased estimator for the variance (an estimator is said to be **unbiased** iff $E(\widehat{\theta}) = \theta$).

The output provided by R through the function var() applied to a sample vector is the sample quantity

$$s^2 = \frac{1}{n-1}(y_i - \bar{y})$$

Leveraging the example above, if we assume that the true generating model is normal (case 1), we know that the distribution of the sample variance is proportional to a χ^2 distribution with $n-1$ degrees of freedom:

$$\frac{(n-1)S^2}{\sigma^2} \sim \chi^2_{n-1}$$

Note
Let X a r.v. with pdf $f_X(x)$, then $Y = g(X)$, with $g(\cdot)$ an invertible function, has pdf

$$f_Y(y) = f_X(g^{-1}(y))\left|\frac{d}{dy}g^{-1}(y)\right|$$

So, in our case let $X = \frac{(n-1)S^2}{\sigma^2} \sim \chi^2_{n-1}$, then $S^2 = Y = \frac{\sigma^2}{n-1}X$ is such that

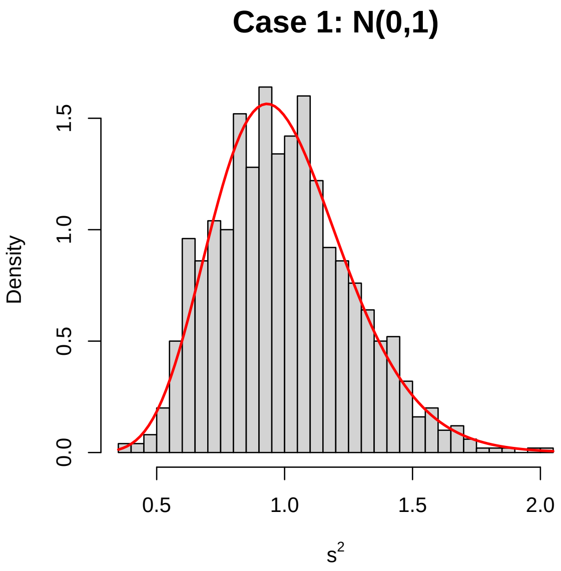
$$f_Y(y) = f_X\left(\frac{n-1}{\sigma^2}y\right)\frac{n-1}{\sigma^2},$$

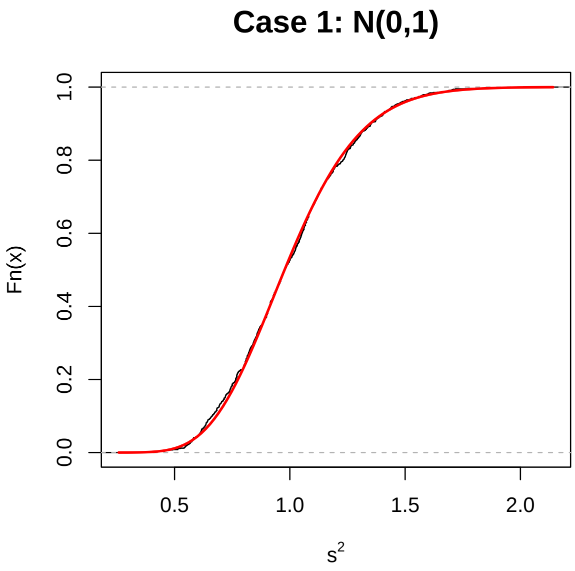
since $g^{-1}(y) = \frac{n-1}{\sigma^2}y$ and $\frac{d}{dy}g^{-1}(y) = \frac{n-1}{\sigma^2}$

Time to work with R: It's your turn

Obtain the distribution of the sample variance via simulation, plot it via histograms/ecd and overlap the corresponding density/cumulative distribution function (using the formula above). After completing the XXX parts, you should obtain the plots in the next page.

```r
par (mfrow = c(1, 2))
sigma <- 1
# Recall the the samples are arleady generated and stored in the array samples
sample_var <- apply(samples, c(1,3), XXX)
# Histogram
hist(XXX, nclass = 30, probability = TRUE,
     xlab = expression(s^2), main = "Case 1: N(0,1)", cex.main = 1.5)
curve(XXX,  add = TRUE, col = "red", lwd = 2)
# ECDF vs CDF
plot(ecdf(XXX), xlab = expression(s^2), main = "Case 1: N(0,1)", cex.main = 1.5)
curve(XXX,  add = TRUE, col = "red", lwd = 2)
```


Case 1: N(0,1)

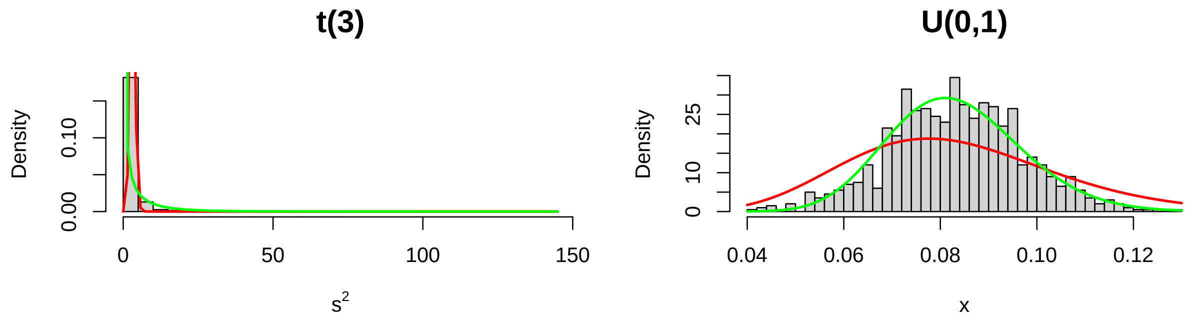

Case 1: N(0,1)

Extra: Sample variance distribution under non Gaussian cases

Consider the following paper https://www.tandfonline.com/doi/abs/10.1080/00031305.2014.966589 (in the Lab2 folder you can find the pdf version). It is useful to understand the distribution of the sample variance $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$ when the Y_i's are not normally distributed.

Thus, we can visualise the distribution of the sample variances for the case 2 and 3 via histograms; we can overlap the probability density function of the sample variance estimators according to the results of the aforementioned paper and compare them with those obtained by considering the distribution of the sample variance for the normal case.

```
par (mfrow=c(1, 2))
sigma <- sqrt(3/(3 - 2))
hist(sample_var[2,],   probability = TRUE,  nclass = 50,
     xlab = expression(s^2), main = "t(3)", cex.main = 1.5)
curve(((n - 1)/sigma^2) * dchisq(x * ((n - 1)/sigma^2), df = n - 1),
      add = TRUE, col="red", lwd=2)
curve(( (2 * sigma^4/var(sample_var[2,]))/sigma^2) *
        dchisq(x *( 2 * sigma^4/var(sample_var[2, ])/(sigma^2)),
        df = 2 * sigma^4/var(sample_var[2,])),
      add = TRUE, col = "green", lwd = 2)

sigma <- sqrt(1/12)
hist(sample_var[3, ], nclass = 50, probability = TRUE,
     xlab = "x", main =  "U(0,1)", cex.main = 1.5)
curve(((n - 1)/sigma^2) * dchisq(x * ((n - 1)/sigma^2), df = n - 1),
add = TRUE, col="red", lwd=2)
curve(( (2 * sigma^4/var(sample_var[3,]))/sigma^2)*
        dchisq(x *( 2 * sigma^4/var(sample_var[3, ])/(sigma^2)),
               df = 2 * sigma^4/var(sample_var[3,])),
      add = TRUE, col = "green", lwd = 2)
```



Basic concepts of estimation

Point estimation

Let $\hat{\theta}$ be the point estimator for the parameter θ. An estimator is said to be

- **unbiased** if and only if $E(\hat{\theta}) = \theta$
- **consistent** if $\hat{\theta} \xrightarrow{P} \theta$, as $n \to \infty$ or, equivalently, if $\text{var}(\hat{\theta}) \to 0$, as $n \to \infty$.

There are some properties that we would ensure for an estimator: **low variance** and **low bias**. But, there is a tradeoff between unbiasedness and low variance, so we would usually seek to get both (to some extent); ideally, we would target a small **Mean Squared Error** (**MSE**)

$$\text{MSE}(\hat{\theta}) = \text{E}\{(\hat{\theta} - \theta)^2\} = \{\text{E}(\hat{\theta}) - \theta\}^2 + \text{var}(\hat{\theta}) = (\text{Bias})^2 + \text{Variance}$$

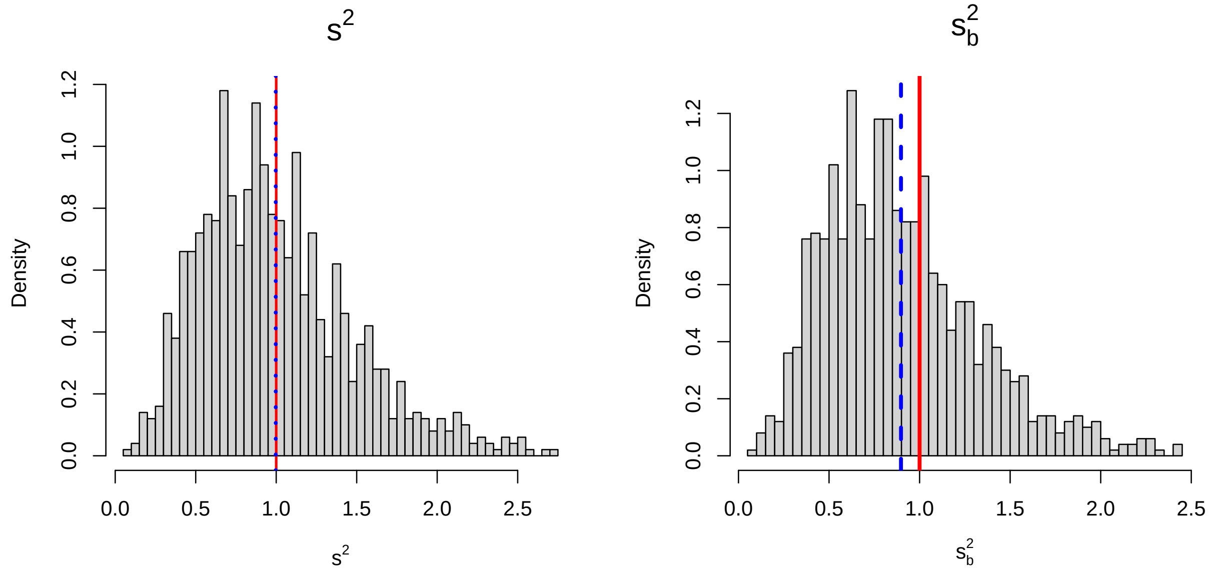
Comparison of unbiased and biased sample variance estimators

Related to the distribution of the sample variance in the normal case, we could also rely on the biased estimator $S_b^2 = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$.

Check the biased nature of S_b^2 via MC simulation, generating $n = 10$ iid values from a normal distribution. Compare the results with the unbiased estimator S^2

```
#Initial settings
set.seed(2)
R <- 1000
n <- 10
mu <- 0
sigma <- 1

# Save the results in two vectors:
# s2: unbiased sample variance estimates
# s2_b: biased sample variance estimates
s2 <- rep(0, R)
s2_b <- rep(0, R)

# For each replication we generate 10 samples from
# a normal r.v. with mean mu and variance sigma^2
# and we compute the four sample variance estimates
for(i in 1 : R) {
  y <- rnorm(n, mu, sigma)
  s2[i] <- var(y)
  s2_b[i] <- var(y) * (n - 1)/n
}
```

```
s2_mean <- mean(s2)
s2_b_mean <- mean(s2_b)

#plot s2
par(mfrow = c(1, 2), oma = c(0, 0, 0, 0))
hist(s2, breaks = 50, xlab = expression(s^2), probability = TRUE,
     main = expression(s^2), cex.main = 1.5)
#in red the true mean, in blue the estimated mean
abline(v = sigma^2, col = "red", lwd = 2)
abline(v = s2_mean, col = "blue", lwd = 3, lty = 3)

#plot s2 biased
hist(s2_b, breaks = 50, xlab = expression(s[b]^2), probability = TRUE,
     main = expression(s[b]^2), cex.main = 1.5)
#in red the true mean, in blue the estimated mean
abline(v = sigma^2, col = "red", lwd = 3)
abline(v = s2_b_mean, col = "blue", lwd = 3, lty = 2)
```



–>