Module 6. Statistical Inferences: Hypothesis testing.

Overview:

This module continues on the topic of statistical inference, discussing hypothesis testing and how to perform hypothesis tests using R. This module consists of several lessons. You will be setting up the hypothesis test, using R (or calculations) with the given data, and interpreting the results. You will work with a variety of tests (one-sided or two-sided) and a variety of distributions. You will also identify various potential sources of errors in hypothesis testing. You will use simulations to calculate the type I and type II error rates of the hypothesis test. Finally, you will learn to adjust the p-values for multiple hypothesis testing. As in previous modules, these lessons include examples, script for how to perform the calculations in R, and opportunities for practice.

Learning Objectives

Hypothesis tests

1. Distinguish between and identify the **null** and **alternative hypotheses**
2. Compare **Type I** and **Type II errors**
3. **Calculate power** using Monte Carlo simulations
4. Distinguish between **one-sided** and **two-sided tests**
5. **Conduct hypothesis tests in R** including one-sample t-test, two-sample t-test, two-sample F-test, and binomial test
6. Using **simulations to verify** a hypothesis test
7. Making adjustments for **multiple testing.**

Readings:

Seefeld & Linder's book pages 240-256.
Krijnen's book Pages 47-58.

Paper "Interval estimation for a binomial proportion" by Lawrence D Brown, T Tony Cai, Anirban DasGupta (2001) Statistical Science pages 101-117. Available at link

http://projecteuclid.org/download/pdf_1/euclid.ss/1009213286

This module introduces the basic concepts of hypothesis testing. Particularly, we consider the one-sample and two-samples test for population means and proportions. We use Monte Carlo simulations for power calculations and checking Type I error rate. Finally, you will be introduced to false discovery rate control in multiple testing

Lesson 1: Hypothesis Testing

Objectives

By the end of this lesson you will have had the opportunity to:

- Distinguish between and identify the null and alternative hypotheses
- Learn the vocabulary of hypothesis testing.
- Learn the relationship between hypothesis testing and confidence intervals
- Derive the t-test for population mean

Overview

Last module covered the parameter estimation. Now we move on to another type of statistical inference: hypothesis testing. This approach follows the common logic in scientific experiments, specifically making decisions based on the numerical evidence in data.

Steps to hypothesis testing

We will first go through the general setup of hypothesis testing. Hypothesis testing frames the statistical inference as making a decision between two opposing claims using evidence from the data set. The process begins with the formulation of a null hypothesis (H_0) and an alternative hypothesis (H_A) from the two opposing claims about the parameter. Generally the alternative is the claim that we wish to prove. We consider the evidence by calculating how likely we are to observe data similar to the given data set, assuming H_0 is correct. This evidence is used to make the decision on these two claims. If the observed data is unlikely to occur under H_0, then we decide that its logic opposition H_A is true.

In summary, the hypothesis testing generally consists of four steps.

(1) Clearly defining the null and alternative hypotheses;

(2) Collect data;

(3) Use the data, to assess the evidence by performing the appropriate test; and

(4) Draw a conclusion based on the test result.

We illustrate the process with an example,

Demonstration: Hypothesis Testing Process

We consider the Gdf5 gene (see Example 1 on page 48 and page 52 of Applied Statistics for Bioinformatics using R)

) If the gene is related to leukemia, its mean expression value μ among ALL patients should be nonzero. Hence, for step (1), we state the null hypothesis H_0: μ = 0 and the alternative hypothesis H_A: μ ≠ 0.

For step (2), we find a data set to test this hypothesis. Here we just use the Golub et al. (1999) data set, and take those Gdf5 gene expression values for the ALL patients.

For step (3), we need to summarize the data and carry out a test. Here the sample mean is $\bar{X}$ = 0.00005. How do we assess the strength of evidence?

This is done by the ***p-value***, the probability of how likely we are to get the observed data under null hypothesis H_0: μ = 0. In this case, the p-value is the probability that we obtain a sample mean value $\bar{X}$ that is at least as far away from the population mean 0 (hypothesized under H_0) as the observed $\bar{X}_{obs}$ = 0.00005. That is, p-value = $P(|\bar{X}-0| \geq |\bar{X}_{obs}-0|) = P(|\bar{X}| \geq 0.00005)$. This calculation can be done using the sampling distribution of the sample mean. Recall, $t = \frac{\bar{X}-\mu}{s/\sqrt{n}}$ follows the t-distribution with degree of freedoms n-1. Hence the observed t-statistic is

$$t_{obs} = \frac{\bar{X}_{obs}-0}{s/\sqrt{n}} = \frac{0.00005-0}{0.259/\sqrt{27}} = 0.001.$$

And the p-value is shown below.

$$P(|t_{df=n-1}| \geq |t_{obs}|) = P(|t_{df=26}| \geq 0.001) = P(t_{df=26} \leq -0.001) + P(t_{df=26} \geq 0.001) = 2P(t_{df=26} \leq -0.001).$$

This can be calculated in R using 2*pt(-0.001,df=26) which results in 0.9992.

Step (4): We draw conclusions based on the evidence in step (3). Since the observed data is highly likely (p-value = 0.9992) under H_0, there is little evidence that the population mean Gdf5 gene expression value is nonzero. Hence, we accept the null hypothesis that the gene is not related to leukemia.

Demonstration (Continued): Using R for Hypothesis Testing

We have seen the four steps in the hypothesis testing. The first two steps involve setting up the hypothesis test, and the last step is the interpretation of the results. Those steps are essential for appropriate data analysis. All the mathematical calculations, however, are done in the step (3). R has programmed functions to conduct those calculations for standard tests. This would free you up to concentrate more on the other three essential steps which always need human judgment.

In step (3) to test H_0 ($\mu = 0$ in this example), we need to calculate an appropriate test statistic (t-statistic), and then calculate the p-value, which is how likely are we to see the observed test statistic. In the previous demonstration, we performed the calculations step by step, where now each step we can use R to calculate: the sample mean $\bar{X}_{obs}$=0.00005, the t-statistic $t_{obs} = \frac{\bar{X}_{obs} - 0}{s/\sqrt{n}} = 0.001$, and the p-value 2*qt(-0.001,df=26).

Alternatively, we can simply use t.test() in R to do **all** these calculations together, without worrying about the details of step by step work. (Load the golub data set from the "multtest" package first. If you forgot how to do this, review Module 2 Lesson 3.)

```
> gol.fac <- factor(golub.cl,levels=0:1, labels= c("ALL","AML"))
> x <- golub[2058,gol.fac=="ALL"]
> t.test(x, mu=0)
        One Sample t-test
data:  x
t = 0.0011, df = 26, p-value = 0.9991
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -0.1024562  0.1025636
sample estimates:
  mean of x
5.37037e-05
```

We can see the p-value = 0.9991 is generated, together with intermediate information of t-statistic = 0.0011, sample mean $\bar{X}_{obs}$=5.37e-05, etc.

Making a Statistical Decision with Hypothesis Testing

We can see in the previous example that the strength of evidence against the null hypothesis is measured by p-value. This p-value describes how surprising our data are when H_o is true. We generally use a cutoff value to decide how small the p-value must be, or how unlikely (or rare) our data must be when H_0 is true, for us to conclude that we have enough evidence to reject H_0. This cutoff is called the **significance level of the test** and is usually denoted by the Greek letter α. The most commonly used significance level is $\alpha = 0.05$ (or 5%). This means that:

- If the p-value $< \alpha$ (usually 0.05), then the data we got is considered to be "rare (or surprising) enough" when H_0 is true, and we say that the data provide significant evidence against H_0, so we reject H_0 and accept H_A.
- If the p-value $> \alpha$ (usually 0.05), then our data are not considered to be "surprising enough" when H_0 is true, and we say that our data do not provide enough evidence to reject H_0 (or, equivalently, that the data do not provide enough evidence to accept H_A).

Wording of the Statistical Decision with Hypothesis Testing

Notice that if the p-value is small ($< \alpha$), we say that the "results are statistically significant". This means that the data provides significant evidence against H_0, thus logically provides evidence for H_A. So the data provides some level of proof for the H_A. We would "reject H_0" and "accept H_A".

When the p-value $> \alpha$, the data does not provides enough evidence against H_0. Logically this does NOT provide evidence for H_0 either; rather it only shows the lack of evidence against H_0. We do often say that we "accept H_0", but that is only because H_0 is the default assumption. A more precise wording is simply that "the results are not statistically significant".

Logically the big p-value does not prove H_0. A small p-value provides some *statistical* proof for H_A. Notice that in statistics nothing is absolute due to randomness. Hence even a small p-value is not a proof (in the strict sense) of H_A. It is rather a probability statement: the evidence from the data supports that H_A is more likely than H_0.

Terminologies and Concepts in Hypothesis Testing

When we perform a hypothesis test, there are four possible outcomes.

	Actual Validity of H_0		
Decision Made		H_0 **is true**	H_0 **is false**
	Accept H_0	True negative	False Negative (Type II Error)
	Reject H_0	False Positive (Type I Error)	True Positive

Two of these outcomes are "correct" decisions – you reject a hypothesis when it should be rejected (true positive), or you accept a hypothesis when it should be accepted (true negative). The other two outcomes are not "correct" but a result of the statistical uncertainty inherent in a hypothesis test. We call these outcomes false results or "errors".

To understand the errors, let us consider the example of testing the existence of gene expression. The null hypothesis is that the gene G is not expressed in the group of patient with disease D. A type I error is the rejection of H_0 given H_0 is true. That is, we declare an unrelated gene G relevant to the disease D: causing a waste of money and effort for creating treatment based on this discovery, and also causing unnecessary stress in gene G carriers if the disease D is serious such as cancer. The type I error rate is the significance level α discussed earlier. If α is 0.05, this means that, when G is unrelated to disease D, we still reject H_0 and declare it relevant 5% of the time.

A type II error is the failure to reject H_0 when H_0 is false. That is, we missed the gene G factor which can lead to potential life-saving treatment of disease D. The type II error rate is often denote by the Greek letter β. The probability of detecting a true positive is 1- β, and is called the **power** of the test.

Hypothesis Testing for Population Mean

The previous example can be considered in a specific setting of testing **H_0:μ = $μ_0$ versus H_A: μ ≠ $μ_0$**, for a random sample $X_1, \ldots, X_n$ from $N(mean = \mu, sd = \sigma)$. We always decide the test under a specified α, the significance level. Then the decision rule should come from the *sampling distributions* under H_0.

Recall from the probability theory, under H_0, $\frac{\bar{X} - \mu_0}{\sigma / \sqrt{n}} \sim N(0,1)$; $\frac{\bar{X} - \mu_0}{s / \sqrt{n}} \sim t_{df = n-1}$.

Hence the p-value is $P(|Z| \geq |Z_{obs}|)$ when σ is known, and $P(|t_{df=n-1}| \geq |t_{obs}|)$ when σ is unknown.

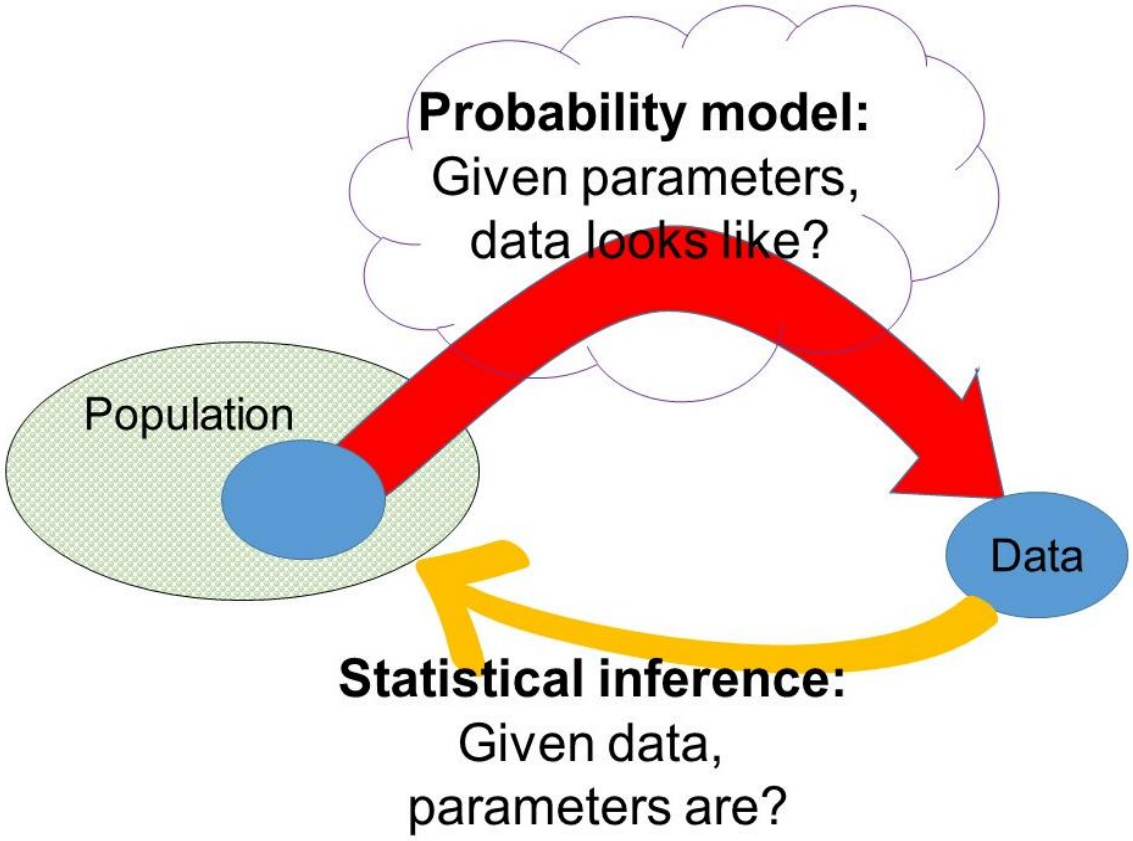
Here $Z_{obs} = \frac{\bar{X}_{obs} - \mu_0}{\sigma / \sqrt{n}}$ and $t_{obs} = \frac{\bar{X}_{obs} - \mu_0}{s / \sqrt{n}}$.

Under the significance level α, we reject H_0:μ = $μ_0$ when **p-value <** α. That is, if $|Z_{obs}| > z_{1-\alpha/2}$ or $|t_{obs}| > t_{1-\alpha/2,n-1}$, where $z_{1-\alpha/2}$ and $t_{1-\alpha/2,n-1}$ denote the 1-α/2 **quantiles** of respectively the standard normal distribution and the t-distribution with degree of freedom n-1.

The same as for confidence intervals: **we always use t-test for real data analysis since σ is unknown.**

Derivation of Hypothesis Testing Procedure

We always conduct statistical inferences based on the probability theory. It is important to remember the overall framework mentioned previously.



For a hypothesis test, the probability theory tells us how likely the data occurs under null hypothesis (this probability is the p-value). We use that to decide between the two opposing hypotheses H_0 and H_A. Similar to the derivation of confidence intervals, the probability (p-value) calculation is based on the sampling distribution.

Confidence Intervals (CI) and Hypothesis Testing for Population Mean

Both derivations comes from the same sampling distribution: for the random sample of $X_1, \ldots, X_n \sim N(\mu,\sigma)$, $\frac{\bar{X}-\mu}{s/\sqrt{n}} \sim t_{df=n-1}$.

The 1-α confidence interval (CI) is $(\bar{X}+t_{\alpha/2,n-1}\frac{s}{\sqrt{n}}, \bar{X}+t_{1-\alpha/2,n-1}\frac{s}{\sqrt{n}})$ because

$P(\bar{X}+t_{\alpha/2,n-1}\frac{s}{\sqrt{n}} < \mu < \bar{X}+t_{1-\alpha/2,n-1}\frac{s}{\sqrt{n}}) = P(t_{\alpha/2,n-1} < \frac{\bar{X}-\mu}{s/\sqrt{n}} < t_{1-\alpha/2,n-1}) = 1-\alpha.$

For testing $H_0: \mu = \mu_0$ versus $H_A: \mu \neq \mu_0$, we reject $H_0: \mu = \mu_0$ when $|t_{obs}| = |\frac{\bar{X}-\mu_0}{s/\sqrt{n}}| \geq t_{1-\alpha/2,n-1}$.

Under null hypothesis, the probability of rejecting H_0 (type I error) is

$P(|\frac{\bar{X}-\mu_0}{s/\sqrt{n}}| \geq t_{1-\alpha/2,n-1}) = 1- P(t_{\alpha/2,n-1} < \frac{\bar{X}-\mu_0}{s/\sqrt{n}} < t_{1-\alpha/2,n-1}) = \alpha.$

Here, rejecting $H_0: \mu = \mu_0$ at α level is equivalent to: μ_0 is not covered by the 1-α confidence interval.

Equivalence of Confidence Interval (CI) and Hypothesis Test

Generally, for any parameter θ, let (θ_L, θ_U) be its 1-α CI. Then we have an α level test for H_0: $\theta=\theta_0$: reject H_0 when θ_0 is not covered by the 1-α CI (θ_L, θ_U).

It is clear that the type I error rate for this test is
$P(\theta_0 \notin (\theta_L, \theta_U)) = 1- P(\theta_0 \in (\theta_L, \theta_U)) = 1- (1-\alpha) = \alpha$.

We can always convert a 1-α confidence interval into an α level hypothesis test, and vice versa. A 1-α confidence interval for θ will contain all those θ_0 values such that H_0: $\theta = \theta_0$ is not rejected at α level.

One-sided Test for Population Mean

In the two-sided test seen previously, the alternative hypothesis is H_A: $\mu \neq \mu_0$. If we only want to prove the population mean is greater than μ_0, then we should set that as the alternative hypothesis in step (1). That is, we test **H_0: $\mu = \mu_0$ versus H_A: $\mu > \mu_0$.**

We always decide on one-sided or two-sided hypothesis test in step (1) by setting the alternative hypothesis H_A as what we *want* to prove.

The p-value for one-sided test is also calculated under the same sampling distribution under H_0. Therefore, the p-value for H_A: $\mu>\mu_0$ is $P(t_{df=n-1} \geq t_{obs})$. Under the significance level α, we reject the null hypothesis and declare the mean is great than μ_0 when $t_{obs} > t_{1-\alpha,n-1}$.

Similarly, to test **H_0: $\mu = \mu_0$ versus H_A: $\mu < \mu_0$,** the p-value is $P(t_{df=n-1} \leq t_{obs})$. Under the significance level α, we reject the null hypothesis and declare the mean is smaller than μ_0 when $t_{obs} < t_{\alpha,n-1}$.

Demonstration: One-sided test for Population Mean.

Suppose we want to show that the mean gene expression values of CCND3 Cyclin D3 in ALL patients is positive (see Example 2 on page 54 of *Applied Statistics for Bioinformatics using R*.) Since we wish to prove $\mu > 0$, the alternative hypothesis is $\mu > 0$. That is, we test H_0: $\mu = 0$ versus H_A: $\mu > 0$.

The data is already collected in the Golub et al. (1999) data set. Recall that the corresponding gene expression values are collected in row 1042 of the golub data matrix (load it if necessary). Hence we can use t.test() in R to test the one-sided alternative of mean **greater** than 0.

```
> t.test(golub[1042,gol.fac=="ALL"],mu=0, alternative = "greater")
        One Sample t-test
data:  golub[1042, gol.fac == "ALL"]
t = 20.0599, df = 26, p-value < 2.2e-16
alternative hypothesis: true mean is greater than 0
95 percent confidence interval:
 1.732853      Inf
sample estimates:
mean of x
 1.893883
```

The p-value is very close to 0 ($<2.2 \times 10^{-16}$). Hence, we reject the null hypothesis and conclude that the gene is indeed positively expressed in ALL patients.

Notice that the one-sided test is done by specifying the value as “greater” or “less” in t.test(). For the two-sided test before, we did not specify “two.sided” since that is the default value.

Lesson Summary

In this lesson, we considered testing for population mean μ. To review, there are four steps in hypothesis testing.

Step (1) Setting up the hypothesis: What you want to prove is the alternative hypothesis. If you want to show the mean is greater (or smaller) than μ_0, use one-sided hypothesis. If you only want to show the mean is different from μ_0, use two-sided hypothesis.

Step (2): Collect the data. The sample mean $\bar{X}$ can be used to test hypothesis for population mean μ.

Step (3): Conduct the t-test (assuming the data is a random sample from the normal distribution). Compute the t-statistic and its p-value under null hypothesis.

Step (4): Decide to reject null hypothesis or not at α significance level (based on a p-value). Make a conclusion.

In the next lesson, we will examine the model assumption and power of t-test in more detail.

Lesson 2: Power Calculation and Model Assumptions for T-test.

Objectives

By the end of this lesson you will have had the opportunity to:

- Calculate power of t-test using R routine
- Calculate power of t-test using Monte Carlo simulations
- Know the model assumptions for t-test

Overview

In this lesson we will consider the calculation of the power of the t-test. Particularly, we will focus on using Monte Carlo method for this calculation. We will also exam the model assumptions of t-test to clarify when it can be used.

Power of the T-test

The **power** of a test is 1- β = P(reject H_0| H_0 is false). Particularly, for an α-level one-sided t-test for H_0: $\mu = \mu_0$ versus H_A: $\mu > \mu_0$, this is P($t_{obs} > t_{1-\alpha,n-1}$ |$\mu > \mu_0$).

This probability is calculated from the sampling distribution of t_{obs} given $\mu > \mu_0$. Notice that $\mu > \mu_0$ contains many μ values and the distribution of t_{obs} varies with μ. Hence, the power is in fact a function of μ: power(μ).

Theoretically, we can find sampling distribution for the t-statistic $t_{obs} = \frac{\bar{X} - \mu_0}{s/\sqrt{n}}$ and then calculate the power at μ. As it is mathematically more advanced, we will use Monte Carlo methods which is computational intensive but very simple to derive.

We do know that the sampling distribution when $\mu = \mu_0$ (null hypothesis) is the t-distribution. Hence, the power converges to α, the significance level, when $\mu \rightarrow \mu_0$.

In fact, the power(μ) function for the one-sided t-test (H_0: $\mu = \mu_0$ versus H_A: $\mu > \mu_0$) has the domain $\mu > \mu_0$. It is a monotone increasing function that starts at power(μ_0) = α. For the two-sided t-test (H_0: $\mu = \mu_0$ versus H_A: $\mu \neq \mu_0$), the domain for power(μ) is $\mu \neq \mu_0$. Also, power(μ_0) = α, power(μ) increases when μ moves away from μ_0 in either direction.

Calculating Power Using Monte Carlo simulation

In order to find the sampling distribution of $t_{obs} = \frac{\bar{X} - \mu_0}{s/\sqrt{n}}$ to do power calculation, we can repeatedly generate the random sample $X_1, \ldots, X_n \sim N(\mu, \sigma = 1)$.

Example: We wish to find the power of the 0.05 level t-test for H_0: $\mu = 3$ versus H_A: $\mu > 3$, when $\mu = 4$, $\sigma = 1$ and the sample size is n = 10.

Solution

Here $\mu_0 = 3$. The t-test rejects H_0 when $t_{obs} = \frac{\bar{X} - 3}{s/\sqrt{10}} > t_{1-\alpha, n-1} = t_{0.95,9}$. To find the power at μ =4, we can generate nsim=10,000 samples of $X_1, \ldots, X_{10}$ from $N(\mu = 4, \sigma = 1)$. The power is estimated by the proportion of samples that rejects H_0. We can use the following in R:

```
> x.sim<-matrix(rnorm(10000*10, mean=4), ncol=10)
> tstat<-function(x) (mean(x)-3)/sd(x)*sqrt(length(x))
> tstat.sim<-apply(x.sim,1,tstat) #Calculate t-test statistic for each data set
> power.sim<-mean(tstat.sim>qt(0.95,df=9)) #Calculate the rejection rate
>  power.sim+c(-1,0,1)*qnorm(0.975)*sqrt(power.sim*(1-power.sim)/10000)
#Display rejection rate (power) with its 95% CI
[1] 0.8873515 0.8934000 0.8994485
```

Thus, the power at $\mu = 4$ is estimated as 89.3%, with its 95% CI as (0.887, 0.899).

Note that we can similarly check the Type I error rate by generating the samples from $N(\mu = 3, \sigma = 1)$ instead, using R

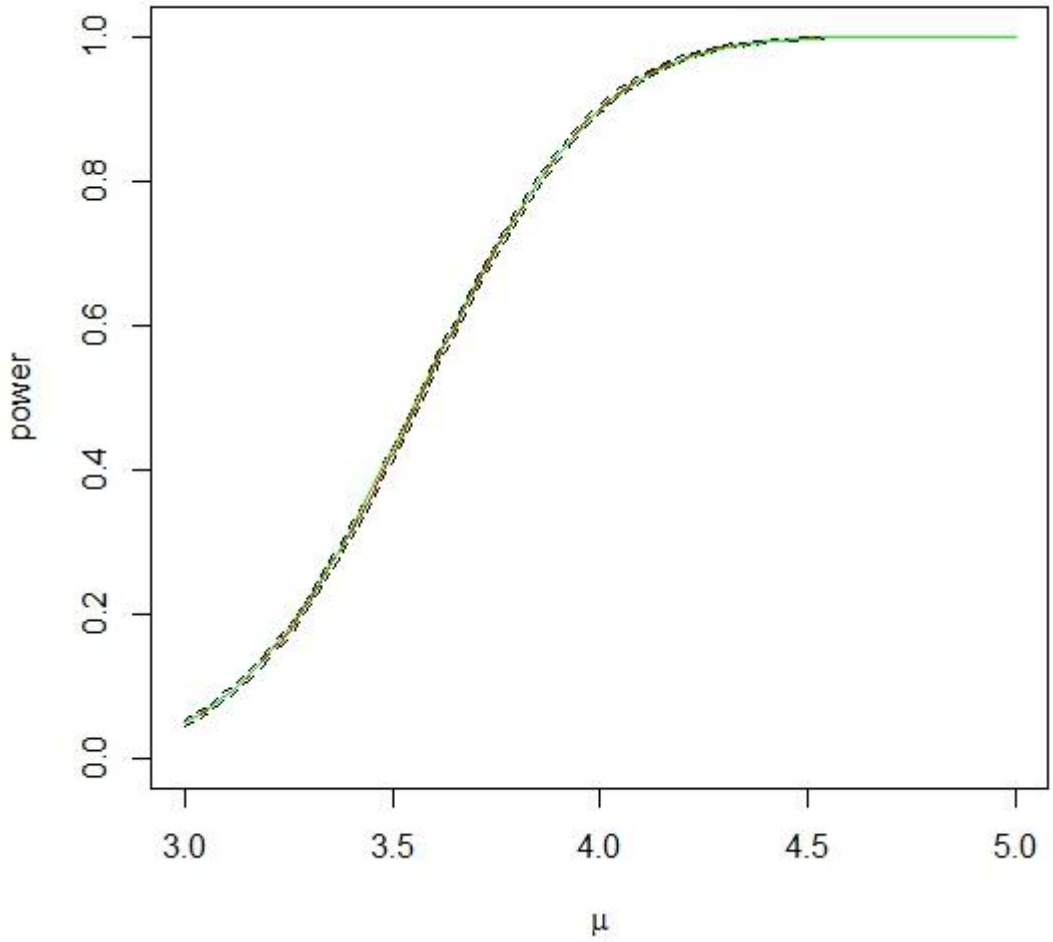
```
> x.sim<-matrix(rnorm(10000*10, mean=3), ncol=10)
> tstat<-function(x) (mean(x)-3)/sd(x)*sqrt(length(x))
> tstat.sim<-apply(x.sim,1,tstat) #Calculate t-test statistic for each data set
> power.sim<-mean(tstat.sim>qt(0.95,df=9)) #Calculate the rejection rate
>  power.sim+c(-1,0,1)*qnorm(0.975)*sqrt(power.sim*(1-power.sim)/10000)
#Display rejection rate (Type I error) with its 95% CI
[1] 0.04419372 0.04840000 0.05260628
```

So the Monte Carlo estimate of the Type I error rate is 0.048 with its 95% CI as (0.044, 0.052). This does agree with the nominal level of $\alpha = 0.05$.

Calculating Power Curve of the t-test

Example (continued) We wish to find the power of the 0.05 level t-test for H_0: $\mu = 3$ versus H_A: $\mu > 3$, when the sample size is $n = 10$.

Recall power(μ) is a function of μ. We can use Monte Carlo method to find the power at various values of μ, and plot them to get the power curve. In this graph, the red line plots the power curve by Monte Carlo simulations.



The dotted lines are the 95% confidence bands for the power from Monte Carlo simulations. The green line is the true power curve calculated using the R function power.t.test(). We see that the red line and green line overlap for most values. So the power found by Monte Carlo simulations are very accurate.

The power of t-test here starts from the 0.05 level ($\mu = 3$), and increases to 1 as true mean μ increases from 3 to 5.

Monte Carlo Calculation of Power Curve

We found the power curve of the one-sided 0.05 level t-test (with sample size n = 10) for H_0: $\mu = 3$ versus H_A: $\mu > 3$, in previous example using Monte Carlo methods.

Here we comment on some computational tricks to reduce the computational time. We can simply just repeat the Monte Carlo power calculation for each value of μ. However, that means for every value of μ, we need to generate 10000 random samples, and calculate the t-statistics on these random samples. That would be very slow and uses a lot memory in R.

We can avoid many repetitions here by noticing the following: if $X_1^*, \ldots, X_n^* \sim N(mean = 0, \sigma = 1)$, then $X_1, \ldots, X_n \sim N(mean = \mu, \sigma = 1)$ for $X_i = X_i^* + \mu$. For these two samples, $\bar{X} = \bar{X}^* + \mu$, $s = s^*$ and

$$t_{obs} = \frac{\bar{X} - \mu_0}{s/\sqrt{n}} = \frac{\bar{X}^* - \mu_0}{s/\sqrt{n}} + \sqrt{n}\mu / s = t^* + \sqrt{n}\mu / s .$$

Therefore, we only need to generate 10000 random samples from the zero-mean normal distribution, and calculate their t-statistics and standard deviations. Then for all other μ values, we can get the new t-statistics with a quick linear transformation $t_{obs} = t^* + \sqrt{n}\mu / s$. This would significantly reduce the computational times. Following is the R code that produces the power curve plot above.

```
x0.sim<-matrix(rnorm(10000*10), ncol=10) #generate data sets (true mean=0)
x0.mean<-apply(x0.sim,1,mean) #Find sample mean for each data set
x0.sd<-apply(x0.sim,1,sd) #Find sample sd for each data set
t0<-(x0.mean-3)/x0.sd*sqrt(10) #Find t-test statistic for each data set
mu.values<-seq(3,5,by=0.01) #mu values where power will be calculated
power.sim<-rep(NA, length(mu.values))
for (i in 1:length(mu.values)) {
    #Calculate power at all mu values
    power.sim[i]<-mean((t0+mu.values[i]/x0.sd*sqrt(10))>qt(0.95,df=9))
}
#Draw the power curve
plot(mu.values,power.sim,type='l', ylim=c(0,1), xlab=expression(mu),
ylab="power",col='red')
#Draw the 95% confidence bounds
ci.lim<-qnorm(0.975)*sqrt(power.sim*(1-power.sim)/10000)
lines(mu.values,power.sim+ci.lim,lty=2)
lines(mu.values,power.sim-ci.lim,lty=2)
```

```
#Compare to real power from power.t.test
lines(mu.values,power.t.test(n=10,delta=(mu.values-3),sig.level=0.05,
type='one.sample', alternative ='one.sided')$power, col='green')
```

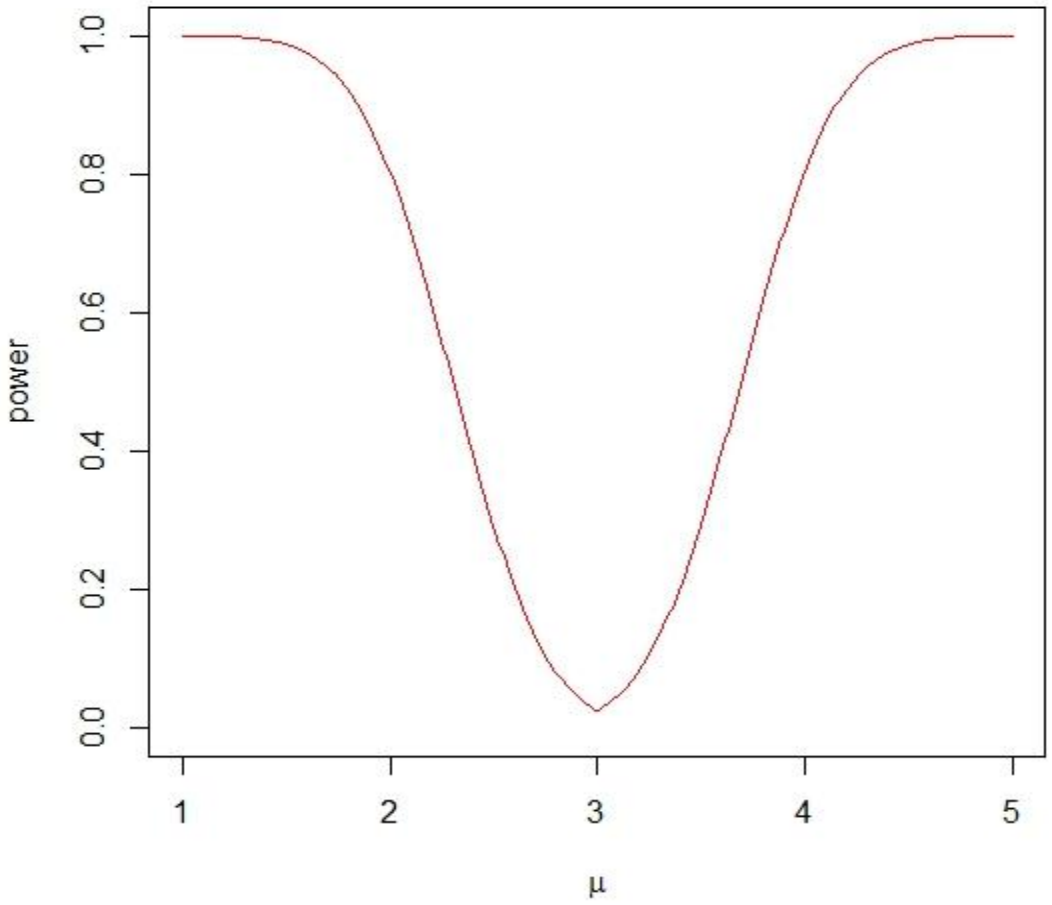
Power Curve of the Two-sided t-test

We have seen the power curve of the 0.05 level t-test (with sample size n = 10) for the one-sided test of H_0: $\mu = 3$ versus H_A: $\mu > 3$. What does the power curve look like for the two-sided test of H_0: $\mu = 3$ versus H_A: $\mu \neq 3$?

We can plot this power curve using R commands

```
mu.values<-seq(1,5,by=0.01)
plot(mu.values, power.t.test(n=10,delta=(mu.values-3),sig.level=0.05,
type='one.sample', alternative ='two.sided')$power, type='l', ylim=c(0,1),
xlab=expression(mu), ylab="power", col='red')
```

In the plot below, we can see that the power of the two-sided t-test starts from the 0.05 level ($\mu = 3$), and increases to 1 as true mean μ moves away from 3 in either direction.



Monte Carlo Power Curve versus Theoretical Curve.

For a t-test, we can calculate its power using the R function power.t.test(). This function is programmed using the theoretical distribution of the t-statistic under alternative hypothesis.

When we have such a theoretical power function, it is clearly more accurate and less time consuming than the Monte Carlo simulation. However, such function requires the theoretical distribution. So someone has to derive the theoretical results and program it. While this was done in R for standard tests such as the t-test, the theoretical distributions for newer and more complex tests are often unknown under the alternative hypothesis.

In contrast, the Monte Carlo simulation is very straightforward for any given test. So it would be very useful when the theoretical results are not available. You would still want to use some simple theoretical properties to reduce the computational time for the Monte Carlo simulation. See the previous example on the one-sided t-test power curve.

When can we use the t-test?

The t-test is derived from the sampling distribution $\frac{\bar{X} - \mu_0}{s/\sqrt{n}} \sim t_{df=n-1}$. Therefore, it is only valid when this t-distribution holds under the null hypothesis.

As we have already seen in probability theory, we derived this sampling distribution in two cases.

(1) The first case is when $X_1, \ldots, X_n$ form a random sample from the normal distribution. So the t-test is valid if we know the data comes from normal distribution.

(2) In the second case, the data is not normally distributed, but the sampling distribution still holds approximately for big sample size n, by the Central Limit Theorem. Therefore, we can apply the t-test for the random sample $X_1, \ldots, X_n$ when n is large, even if the random sample does not come from normal distribution.

Notice that for non-normal data with small sample size, we do not have the t-distribution as the sampling distribution. Therefore, we cannot rely on the t-test for small sample unless we are sure about the normality of the data. For a small sample size, the nonparametric tests will be a safer choice than t-test. We will cover the nonparametric tests in a later module.

Example: T-test on Chi-square Data

Suppose that we have a random sample $X_1, \ldots, X_{100}$ from the Chi-square distribution. We use a 0.05 level two-sided t-test for H_0: $\mu = 13$ versus H_A: $\mu \neq 13$. Is this a valid test? What is its power when $\mu = 15$?

We can find the Type I error rate through Monte Carlo simulations.

```
> x.sim<-matrix(rchisq(10000*100, df=13), ncol=100)
> p.sim<-apply(x.sim, 1, function(x) t.test(x, mu=13)$p.value)
> power.sim<- mean(p.sim<0.05)
>
power.sim+c(-1,0,1)*qnorm(0.975)*sqrt(power.sim*(1-power.sim)/10000)
[1] 0.04668812 0.05100000 0.05531188
```

The estimated Type I error rate is 0.051 with its 95% CI (0.047, 0.055). So this t-test is valid here.

The power at $\mu = 15$ can be similarly computed with random samples from Chi-square distribution with degree of freedom df = 15.
The estimated power at $\mu = 15$ is 0.963 with its 95% CI (0.960, 0.967).
Notice that the power.t.test() would not give correct power in this case, since the data is **not normally** distributed.

For large sample size n, the t-test remains valid here due to the Central Limit Theorem. This is confirmed by the above simulation.

However, we can check that for a small sample size n = 10, the Type I error rate is no longer correct. A Monte Carlo estimate for Type I error rate is 0.059 with its 95% CI (0.055, 0.064). Therefore, the t-test's Type I error rate exceeds its nominal 0.05 level when n = 10.

Notes on the Previous Example

We used Monte Carlo simulations to study the Type I error rate and power for t-test on data from Chi-square distribution.

If we do know that the data comes from Chi-square distribution, then we should use the t-statistic $\frac{\bar{X}-\mu_0}{\sqrt{2\bar{X}}/\sqrt{n}}$ (since chi-square variance = 2*d.f. = 2*mean) instead of the $\frac{\bar{X}-\mu_0}{s/\sqrt{n}}$ before. However, in practice, when we see a data set, we do not know the underlying distribution. So the usual t-test would be used.

The Monte Carlo study is a theoretical sensitive analysis (for the normality assumption) on the practical t-test using $\frac{\bar{X}-\mu_0}{s/\sqrt{n}}$. It confirms what Central Limit Theorem tells us: the t-test is valid for non-normal data when the sample size is large.

In practice, we can safely apply the t-test in large sample size.

Summary

In this lesson, we focused on how to calculate the power and significance level for t-test. The R function power.t.test() would calculate these quantities. However, you should know how to find these quantities through Monte Carlo simulation. The later approach can be generally applied to all other tests.

The t-test is based on the sampling distribution (a t-distribution), which is true for normally distributed data or is approximately true for large sample size. So t-test should only be used in those situations. For small sample size, nonparametric tests should be used.

This lesson considers t-test for one population mean. The next lesson will be on tests for comparing two population means.

Lesson 3: Two-sample t-test

Objectives

By the end of this lesson you will have had the opportunity to:

- Distinguish between paired and unpaired two-sample t-tests
- Conduct two-sample t-tests with R

Overview

We have seen the t-test on a random sample for testing the population mean. Sometimes, we have two random samples from two populations respectively, and would like to compare the two population means. This is often done by two-sample t-test.

Paired t-test

Suppose that we have a random sample $X_1, \ldots, X_n$ from $N(\mu_X, \sigma_X)$ and a random sample $Y_1, \ldots, Y_n$ from $N(\mu_Y, \sigma_Y)$. We would like to know if the population means μ_X and μ_Y in these two groups are the same. That is, we test H_0: $\mu_X = \mu_Y$ versus H_A: $\mu_X \neq \mu_Y$.

Notice that if we pair the Xs and Ys, and taking their pairwise differences, we obtain $D_1 = X_1 - Y_1, \ldots, D_n = X_n - Y_n$ as a random sample from $N(\mu_D = \mu_X - \mu_Y, \sigma_D)$. Then the two-samples mean comparison reduces to a one-sample hypothesis test on Ds as H_0: $\mu_D = \mu_X - \mu_Y = 0$ versus H_A: $\mu_D \neq 0$. The one-sample t-test in previous lessons can be used directly.

For data analysis in R, the data generally are stored in two vectors x and y. We can calculate d = x-y and then apply the t.test() on the vector d as before. R also can do the paired t-test directly using t.test(x,y, paired=T).

Demonstration: Paired t-test on Two Genes

Example. Supposewe want to show that two genes (CD33 and CCND3 Cyclin D3) express differently, using the Golub data set. That is, we test H_0: $\mu_X = \mu_Y$ versus H_A: $\mu_X \neq \mu_Y$, where μ_X and μ_Y denote the mean gene expression values for the CD33 gene and the CCND3 Cyclin D3 gene respectively.

The two genes are stored in the 808th row and 1042th row. We can use R to calculate:

```
> t.test(golub[808,],golub[1042,],paired=T)

        Paired t-test

data:  golub[808, ] and golub[1042, ]
t = -9.2303, df = 37, p-value = 3.886e-11
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -2.504094 -1.602609
sample estimates:
mean of the differences
              -2.053352
```

We can see that the two genes do express differently (p-value = 0). From the confidence interval (-2.5,-1.6), the CD33 gene clearly has lower mean expression value than the CCND3 Cyclin D3 gene.

Demonstration (Continued)

We can also calculate the pairwise differences and then apply the one-sample t-test. It produces essentially the same output in R as the output of using paired t-test directly.

```
> t.test(golub[808,]-golub[1042,])

        One Sample t-test

data:  golub[808, ] - golub[1042, ]
t = -9.2303, df = 37, p-value = 3.886e-11
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -2.504094 -1.602609
sample estimates:
mean of x
-2.053352
```

Two-sample t-test (Unpaired)

Suppose that we have a random sample $X_1, \ldots, X_n$ from $N(\mu_X, \sigma_X)$ and a random sample $Y_1, \ldots, Y_m$ from $N(\mu_Y, \sigma_Y)$, with **different sample sizes n and m.** Then we cannot use the paired test anymore for testing H_0: $\mu_X = \mu_Y$ versus H_A: $\mu_X \neq \mu_Y$.

Naturally we would reject H_0: $\mu_X - \mu_Y = 0$ when the statistic $|\hat{\mu}_X - \hat{\mu}_Y| = |\bar{X} - \bar{Y}|$ is big. Using the properties on linear combination of random variables, $\bar{X} - \bar{Y}$ has mean $\mu_X - \mu_Y$ and variance $\frac{\sigma_X^2}{n} + \frac{\sigma_Y^2}{m}$ (**assuming independence between Xs and Ys**), and it is normally distributed. That is, under H_0, $\bar{X} - \bar{Y} \sim N(0, \sqrt{\frac{\sigma_X^2}{n} + \frac{\sigma_Y^2}{m}})$.

Since σ_X and σ_Y are unknown, we need the test statistic $\frac{\bar{X} - \bar{Y}}{\sqrt{\frac{s_X^2}{n} + \frac{s_Y^2}{m}}}$ instead.

Generally the sampling distribution of this statistic is not known exactly. However, $\frac{\bar{X} - \bar{Y}}{\sqrt{\frac{s_X^2}{n} + \frac{s_Y^2}{m}}}$ approximately follows a t-distribution with fractional degree of freedom. The formula is rather complicated, but it is programmed into R.

The t-test based on this test statistic is called the **Welch two-sample t-test**.

When the data are saved in two vectors x and y, we can just use **t.test(x,y)** for the Welch two-sample t-test.

Two-sample t-test

Example We want to show that the CCND3 Cyclin D3 gene expresses differently in ALL patients from in AML patients, using the Golub data set. That is, we test H_0: $\mu_X = \mu_Y$ versus H_A: $\mu_X \neq \mu_Y$, where μ_X and μ_Y denote the mean CCND3 Cyclin D3 gene expression values in ALL patients and AML patients respectively.

```
> data(golub, package = "multtest")
> gol.fac <- factor(golub.cl,levels=0:1, labels= c("ALL","AML"))
> t.test(golub[1042, gol.fac=="ALL"], golub[1042, gol.fac=="AML"] )

        Welch Two Sample t-test

data:  golub[1042, gol.fac == "ALL"] and golub[1042, gol.fac == "AML"]
t = 6.3186, df = 16.118, p-value = 9.871e-06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.8363826 1.6802008
sample estimates:
mean of x mean of y
1.8938826 0.6355909
```

The p-value is extremely small. We reject the null hypothesis and conclude that the gene do express in ALL patients differently from in AML patients.

More R commands

We see that t.test(x, y) conducts the two-sample t-test on the two groups of data saved in vectors x and y. Sometimes, the observations Xs and Ys are all saved in one (column) variable 'x', and another variable 'group' contains their group indicator. In that case, instead of separating the Xs and Ys according to the two groups before t-test, we can also directly do t.test(x~group). This command does the two-sample t-test between the two groups indicated by the 'group' variable.

In previous example, 'gol.fac' is such a group indicator for the ALL and AML groups. Hence we can also do t.test(golub[1042,]~ gol.fac) instead. This gives exactly the same outputs which we got earlier with command t.test(golub[1042, gol.fac=="ALL"], golub[1042, gol.fac=="AML"]).

Two-sample t-test (Equal Variances)

The two-sample t-test is based on an approximate sampling distribution of the test statistic $\frac{\bar{X}-\bar{Y}}{\sqrt{\frac{s_X^2}{n}+\frac{s_Y^2}{m}}}$. We can get an exact t-distribution test statistic when $\sigma_X = \sigma_Y$.

Let $s_p^2 = \frac{(n-1)s_X^2+(m-1)s_Y^2}{n+m-2}$ be the pooled variance estimator combining the two groups together. Then $\frac{\bar{X}-\bar{Y}}{s_p^2\sqrt{\frac{1}{n}+\frac{1}{m}}} \sim t_{df=n+m-2}$ under the null hypothesis.

Then the two-sample t-test can be conducted based on this sampling distribution. The two-sample t-test with equal variances in R is done with t.test() by specifying the 'var.equal=T' option.

Example (continued). Assuming the variances of the CCND3 Cyclin D3 gene expression values are the same in the ALL and AML groups, the two-sample t-test for their mean (using R) is the following:

```
> t.test(golub[1042,]~ gol.fac, var.equal=T )

        Two Sample t-test

data:  golub[1042, ] by gol.fac
t = 6.7983, df = 36, p-value = 6.046e-08
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.8829143 1.6336690
sample estimates:
mean in group ALL mean in group AML
        1.8938826         0.6355909
```

This small p-value also leads us to conclude that the mean gene expression values are different in the two groups.

Two-sample t-test

We have two versions of the two-sample t-tests, one for unequal variances and one for equal variances. How do we decide which one to use in practice? Since we generally do not know beforehand if the variances are equal, we should always prefer the Welch two-sample t-test with unequal variances.

If we want to use the two-sample t-test with equal variances, we need to test the hypothesis of equal variances first.

F-test for Equal Variances

Suppose that we have two random samples $X_1, \ldots, X_n$ from $N(\mu_X, \sigma_X)$ and $Y_1, \ldots, Y_m$ from $N(\mu_Y, \sigma_Y)$. We want to test **H_0: $\sigma_X = \sigma_Y$ versus H_A: $\sigma_X \neq \sigma_Y$.**

Since s_X and s_Y are sample estimates for σ_X and σ_Y, the natural test statistic here is their ratio. We reject the null hypothesis if the ratio s_X/s_Y is far away from 1, either too big or too small.

From probability theory, under the null hypothesis, the sampling distribution of $\frac{s_X^2}{s_Y^2}$ is the F-distribution with degrees of freedoms $df_1 = n-1$ and $df_2 = m-1$.

Therefore, we reject H_0: $\sigma_X = \sigma_Y$ when $\frac{s_X^2}{s_Y^2} < F_{\alpha/2,n-1,m-1}$ or $\frac{s_X^2}{s_Y^2} > F_{1-\alpha/2,n-1,m-1}$. This F-test is programmed in R as the var.test().

For the previous example on testing the CCND3 Cyclin D3 gene expressions, we have the following using R:

```
> var.test(golub[1042,]~ gol.fac)

        F test to compare two variances

data:  golub[1042, ] by gol.fac
F = 0.7116, num df = 26, denom df = 10, p-value = 0.4652
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.2127735 1.8428387
sample estimates:
ratio of variances
         0.7116441
```

The large p-value 0.465 leads us to accept the equal variances null hypothesis. Therefore, the two-sample t-test with equal variances can be used here.

Summary:

The two-sample t-test tests the equivalence of two groups means, based on data of two random samples $X_1, \ldots, X_n$ from $N(\mu_X, \sigma_X)$ and $Y_1, \ldots, Y_m$ from $N(\mu_Y, \sigma_Y)$.

The paired t-test can be used if the sample sizes are the same n = m, and is done in R using t.test(x,y, paired=T). The paired t-test does **NOT** require the independence between the two samples. It is also more powerful than the unpaired t-test.

The general (unpaired) Welch two-sample t-test is done in R using t.test(x,y) or t.test(x~group).

There is also an (unpaired) two-sample t-test with equal variances, done in R using t.test(x,y, var.equal=T) or t.test(x~group, var.equal=T). This t-test can be used only if the F-test var.test(x,y) or var.test(x~group) does not reject the equal variances assumption. Generally, we should use the Welch two-sample t-test without equal variance assumption, just to be safe.

Next lesson will be on testing population proportions, in contrast to the population means tests in this lesson.

Lesson 4: Testing Population Proportions

Objectives

By the end of this lesson you will have had the opportunity to:

- Conduct one-sample z-test and two-sample z-test for population proportions in R
- Conduct the exact binomial test for one-sample population proportion in R
- Distinguish when to apply one-sample versus two-sample tests, when to use binomial test.

Overview

In this lesson, we consider the hypothesis tests for population proportions. Similar to the testing of population means, these tests include one-sided test, two-sided test, one-sample tests and two-sample tests.

The point estimator for the population proportion is the sample proportion. The inference uses the sampling distribution of the sample proportion, which is related to the Binomial distribution. As covered in the earlier module on confidence intervals, this sampling distribution is approximately normal for large sample size n, according to the Central Limit Theorem.

Hypothesis test for One Population Proportion

If we want to show that the proportion (p) of a certain type A in the population is not p_0, this becomes a hypothesis test H_0: $p = p_0$ versus H_A: $p \neq p_0$.

For a random sample of size n from this population, the number of type A in the sample (X) follows a Binomial(size = n, prob = p) distribution. The point estimator for p is the sample proportion $\hat{p} = X / n$. Under the null hypothesis H_0, X~ Binomial(size = n, prob = p_0). By the Central Limit Theorem, the sampling distribution of $\hat{p}$ under H_0 is approximately N(mean= p_0, sd= $\sqrt{\frac{p_0(1-p_0)}{n}}$).

Therefore, we reject H_0: $p=p_0$ when $\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}} < z_{\alpha/2}$ or $\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}} > z_{1-\alpha/2}$.

If we want to show the population proportion is less than p_0, then this becomes a one-sided hypothesis testing H_0: $p = p_0$ versus H_A: $p < p_0$. Based on the sampling distribution, we reject H_0: $p = p_0$ when $\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}} < z_{\alpha}$.

In R, these tests for the proportion are coded as prop.test().

Demonstration

Suppose an experiment crossing flowers of two genotypes produce progeny with white flowers (recessive) and progeny with purple flowers (dominant). We want to test if the crossing is random (which produces 1/4 proportion of recessive progeny and 3/4 dominant progeny). Then this becomes a hypothesis test of H_0: p = 3/4 versus H_A: p ≠ 3/4 for the proportion p of purple flowers.

We have empirical data from 900 plants, 625 of which have purple flowers, and the remainder 275 have white flowers. Hence $\hat{p} = 625/900 = 0.694$. The test statistic $Z_{obs} = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}} = \frac{0.694 - 0.75}{\sqrt{0.75(1-0.75)/900}} = -3.85$. Hence the p-value is 0.000118 (calculated using 2*pnorm(-3.85)), and we reject the null hypothesis. Therefore, this data does not conform to the assumed 3/4 proportion of recessive progeny (under simple Mendelian inheritance). The reason of the violation needs further investigation asthe crossing that produced the 900 plants may not be random.

Demonstration: (Continued) Using prop.test()

In the previous example, we calculated by hand $Z_{obs} = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}} = \frac{0.694 - 0.75}{\sqrt{0.75(1-0.75)/900}} = -3.85$ and the p-value 0.000118. We can also apply prop.test() in R.

```
> prop.test(x=625,n=900,p=3/4, alternative="two.sided",    correct=F)

        1-sample proportions test without continuity correction

data:  625 out of 900, null probability 3/4
X-squared = 14.8148, df = 1, p-value = 0.0001186
alternative hypothesis: true p is not equal to 0.75
95 percent confidence interval:
 0.6635759 0.7236601
sample estimates:
      p
0.6944444
```

Notice that the prop.test() used the test statistic $(Z_{obs})^2 = (-3.849)^2 = 14.8148$ instead. Recall from probability theory, the Chi-square distribution with degree of freedom 1 comes from the square of standard normal random variable $Z^2 \sim [N(0,1)]^2 = \chi^2_{df=1}$. $(z_{1-\alpha/2})^2 = \chi^2_{1-\alpha,df=1}$ so that the two-sided z-test is equivalent to the chi-square test.

The default option is 'correct=T' in prop.test(), which would apply a continuity correction on $(Z_{obs})^2$ with $(Z_{obs})^2 = (\frac{|\hat{p} - p_0| - 1/2n}{\sqrt{p_0(1-p_0)/n}})^2$. This compensates the fact that our observation is integer but we are approximating with a continuous (normal) distribution.

Binomial Test (The Exact Test for Proportion)

The proportion test above is based on the approximate sampling distribution of $\hat{p} = X / n$ as N(mean= p_0, sd= $\sqrt{\frac{p_0(1-p_0)}{n}}$) under the null hypothesis.

However, we know the exact distribution of X under the null hypothesis is Binomial(size=n, prob=p_0). Hence the p-value can be calculated directly from this binomial distribution. In R, this is programmed as binom.test().

For the previous example, we can do the following.

```
> binom.test(x=625, n=900,p=3/4, alternative="two.sided")

        Exact binomial test

data:  625 and 900
number of successes = 625, number of trials = 900, p-value = 0.0001593
alternative hypothesis: true probability of success is not equal to 0.75
95 percent confidence interval:
 0.6631931 0.7244169
sample estimates:
probability of success
             0.6944444
```

We can see that the point estimator is exactly the same $\hat{p} = 625/900 = 0.694$. The p-value is very close to that from prop.test(), due to the large sample size.

In practice, we would always want to use the exact test (binom.test here). We do cover the prop.test as it is based on the approximate normal sampling distribution, which is also used to derive the two-sample proportion test.

Asymptotic Test Versus Exact Test

Asymptotic test is based on the asymptotic approximated distribution of test statistic. That is, the distribution is approximately true for large sample size. The z-test (prop.test here) is an asymptotic test, which can only be used for large n.

The **exact** test is based on exact distribution of the test statistic. The binomial test here is an exact test. We always prefer the exact test over the asymptotic test. However, sometimes exact test may be hard to get or too computationally intensive.

Demonstration: Binomial test

A researcher believes that more than 70% of bases in a certain type of micro RNA are purines. To prove the claim, we need to set this up as one-sided hypothesis test of H_0: p = 0.7 versus H_A: p > 0.7.

Suppose one micro RNA of this type has length 22 and contains 18 purines. Does the data support the claim?

Solution

We can compute the p-value from the binomial distribution by

P(X≥18)= 1-pbinom(17,size=22,prob=0.7)=0.1645.

This test can also be conducted by the function binom.test() as follows.

```
> binom.test(x=18,n=22,p=0.7, alternative="greater")

        Exact binomial test

data:  18 and 22
number of successes = 18, number of trials = 22, p-value = 0.1645
alternative hypothesis: true probability of success is greater than 0.7
95 percent confidence interval:
 0.6309089 1.0000000
sample estimates:
probability of success
             0.8181818
```

The *p*-value 0.1645 is large. Thus, we cannot reject the null hypothesis at the significance level 0.05.

Two proportions Comparison

We discussed how to test if one population proportion equals a value p_0. Now we consider testing if two proportions p_1 and p_2 are the same.

The null hypothesis here is H_0: $p_1 = p_2$. The alternative hypothesis is the claim that we wish to prove. It can be the two-sided H_A: $p_1 \neq p_2$, or the one-sided H_A: $p_1 > p_2$ or H_A: $p_1 < p_2$.

For the two independent random samples X_1~Binomial(size = n_1, prob = p_1) and X_2~Binomial(size = n_2, prob = p_2), the point estimator for p_1 and p_2 are respectively $\hat{p}_1 = \frac{X_1}{n_1}$ and $\hat{p}_2 = \frac{X_2}{n_2}$. When the sample sizes are large, the approximate sampling distribution of $\hat{p}_1 - \hat{p}_2$ is $N(mean = p_1 - p_2, sd = \sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}})$. Under null hypothesis H_0: $p_1 = p_2$, the standard deviation is estimated by $\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_1} + \frac{1}{n_2})}$.

Here $\hat{p}$ is the "pooled" estimator of the common proportion $\hat{p} = \frac{X_1 + X_2}{n_1 + n_2}$.

Then the test should be conducted based on the z-test statistic $Z_{obs} = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_1} + \frac{1}{n_2})}}$ whose sampling distribution under null hypothesis is approximately $N(mean = 0, sd = 1)$. Therefore, the null hypothesis will be rejected if the z-test statistic is **too far away from zero**.

The test is programmed into R function prop.test().

Demonstration

We wish to know if the proportions of purines in two regions of the genome are the same. Hence this is a two-sided hypothesis test for $\mathbf{H_0: p_1 = p_2}$ **versus** $\mathbf{H_A: p_1 \neq p_2}$.

Two sequences, one from each of the two regions, have lengths 34 and 28 respectively with 17 and 16 purines. What do we conclude?

Solution

We do the test by prop.test() in R:

```
> prop.test(x=c(17,16), n=c(34,28), alternative="two.sided")

        2-sample test for equality of proportions with continuity correction

data:  c(17, 16) out of c(34, 28)
X-squared = 0.0932, df = 1, p-value = 0.7602
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.3526777  0.2098206
sample estimates:
   prop 1    prop 2
0.5000000 0.5714286
```

The *p*-value 0.76 is large. We cannot reject the null hypothesis. The data does not provide evidence that the purine proportions are different in the two regions of genome.

If we do the test by hand, then $\hat{p}_1 = \frac{17}{34} = 0.5$, $\hat{p}_2 = \frac{16}{28} = 0.571$, $\hat{p} = \frac{17+16}{24+28} = 0.532$ so that the z-test statistic $Z_{obs} = \frac{0.5 - 0.571}{\sqrt{0.532(1-0.532)(\frac{1}{34}+\frac{1}{28})}} = -0.561$. This would agree with the prop.test() *without continuity correction*.

prop.test(x=c(17,16), n=c(34,28), alternative="two.sided", correct=F) would give the test statistic $(Z_{obs})^2 = (-0.561)^2 = 0.3147$ with p-value=0.5748.

We would want to use prop.test() instead of doing these calculations by hand. The default option *with continuity correction* is better than *without continuity correction.*

Confidence Intervals (CI) and Hypothesis Test for the Proportion

We have shown that a 1-α CI (θ_L, θ_U) is equivalent to an α level test for H_0: $\theta=\theta_0$ which rejects H_0 when $\theta_0 \notin (\theta_L, \theta_U)$.

For the population proportion, the Ward's two-sided 1-α CI formula $(\hat{p} + z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}})$ is based on the statistic $\frac{\hat{p}-p}{\sqrt{\hat{p}(1-\hat{p})/n}}$. In contrast, the z-test for H_0: $p=p_0$ is based on $\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}}$. Here, rejecting H_0: $p=p_0$ by the z-test does not exactly corresponds to $p_0 \notin (\hat{p} + z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + z_{1-\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}})$. This is due to the use of different sampling distributions $\frac{\hat{p}-p}{\sqrt{\hat{p}(1-\hat{p})/n}}$ versus $\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}}$.

In fact, we can get a 1-α CI for p by including all values p_0 such that H_0: $p=p_0$ is not rejected by the z-test. That is, solve $|\frac{\hat{p}-p_0}{\sqrt{p_0(1-p_0)/n}}| < z_{1-\alpha/2}$ for p_0 value. This results in the Wilson's interval $\frac{\hat{p}+(z_{\alpha/2})^2/n}{1+(z_{\alpha/2})^2/n} \pm \frac{z_{\alpha/2}}{1+(z_{\alpha/2})^2/n}\sqrt{\frac{\hat{p}(1-\hat{p})+(z_{\alpha/2})^2/4n}{n}}$.

Notice that prop.test() also gives a confidence interval which is a variation of the Wilson's interval: solving the z-test with continuity correction $|\frac{|\hat{p}-p_0|-1/2n}{\sqrt{p_0(1-p_0)/n}}| < z_{1-\alpha/2}$.

Summary

In this lesson, we taught the one-sample and two-sample tests for population proportions. The tests are based on the sampling distribution, a normal distribution by approximation from the Central Limit Theorem. In practice, it is better to use the prop.test() function which by default does the continuity correction.

For one-sample proportion test, binom.test() does the exact Binomial test. An exact test should be used whenever the computational resource allows. It gives more accurate result without requirements such as large sample size for the Central Limit Theorem.

Reversing the hypothesis test would give better confidence interval for proportion than the conventional Ward's confidence interval. It is recommended to use the confidence intervals given by prop.test() and binom.test() in practice.

In the next and final lesson, we teach the method for adjusting p-values to control error rates for multiple hypothesis testing. That is extremely important in bioinformatics applications.

Lesson 5: Multiple Testing Adjustment

Objectives

By the end of this lesson you will have had the opportunity to:

- Conduct hypothesis testing with controlled false discovery rate

Overview

In a typical microarray experiment, often a huge number (10,000s to 100,000s) of statistical hypotheses are tested on the same data set. We have been deriving tests at a given significant level α. Even if the significance level α is small, many false positives are bound to occur when a big number of tests are done. For example, if we test 1000 gene expressions at the $\alpha = 0.05$ level, on average we will find 50 "expressed" genes even if all the genes are not related to the disease (all 1000 null hypotheses are true).

To control the rate of these false discoveries, multiple testing adjustments are needed. A simple method, called the **Bonferroni adjustment**, is to reject the null hypothesis only for p-values less than α/m (instead of α) when testing m hypothesis. This will ensure that P(at least one false rejection of null hypothesis) $\leq \alpha$. However, this adjustment results in very low power for the testing. We will introduce procedures that control the **False Discovery Rate (FDR**) instead.

Error Rates:

Assume we are testing m hypotheses H_0^1, H_0^2, …, H_0^m.
Denote m_0 = # of true null hypotheses, R = # of rejected hypotheses

	Null True	Alternative True	Total
No Rejection	U	T	m-R
Rejection	V	S	R
	m_0	$m- m_0$	m

Then V = # Type I errors (false positives). Then type I error rates can be extended to the multiple testing situations in different ways, including the following two:

- **Family-wise error rate (FWER)** is the probability of at least one false positive.

FEWR = P(V ≥ 1)

- **False discovery rate (FDR)** is the expected proportion of Type I errors among the rejected hypotheses

FDR = E(V/R) where we take convention 0/0=0 when R=0.

The **Bonferroni adjustment** controls **Family-wise error rate (**FEWR). FEWR controls the false discovery proportion in all tested hypotheses while FDR controls only the proportion in all rejected hypotheses. These two are the same when all null hypotheses are true ($m_0 = m$). But when there are true discoveries ($m_0 < m$), FEWR ≥ FDR. Controlling FDR will lead to more powerful tests, making more discoveries.

Controlling False Discovery Rate (FDR) with R

We can use the package p.adjust() to make multiple testing adjustments to the p-values, including the Bonferroni correction ("bonferroni") and Benjamini & Hochberg (1995)'s method to control FDR ("BH" or its alias "fdr").

We illustrate the usage of p.adjust() through the following example.

Example : We consider all genes in the Golub et al. (1999) data set. We apply the t-test to each gene to check if it is expressed. The p-value of the t-tests are then adjusted in R by Bonferroni and FDR.

```
> data(golub, package = "multtest")
> p.values <- apply(golub, 1, function(x) t.test(x)$p.value)
> p.bon <-p.adjust(p=p.values, method="bonferroni")
> p.fdr <-p.adjust(p=p.values, method="fdr")
> sum(p.values<0.05)
[1] 2529
> sum(p.bon<0.05)
[1] 1827
> sum(p.fdr<0.05)
[1] 2512
```

There are in total 3051 genes in the data set. 2529 of these have p-values < 0.05. Bonferroni correction results in 1827 significantly expressed genes with p-values $< 0.05/3051 = 1.62 \times 10^{-5}$. As you can see, many genes are indeed expressed.

Using the FDR adjustment, more genes (2512 genes) are declared to be expressed significantly than using the Bonferroni correction.

P-values and Q-values

The outputs of p.adjust() are often called as the "adjusted p-values". The Bonferroni correction (m*p-value) lead to a bound on the family-wise Type I error rate, so that is an "adjusted p-value". The **false discovery rate (**FDR) correction does not control the Type I error rate, but the false discovery rate. To distinguish, that is called q-value and is not really a p-value. So the "adjusted p-value" is in fact a misnomer for the FDR adjustment.

p.adjust(p, method="fdr") changes the p-value of individual test into the **q-value**. For q-value (false discovery rate bound), we do not need to stick with the conventional 0.05 level. Particularly for sparse discoveries ($m_0 \approx m$), bigger q-value cutoffs (such as 0.5) still make practical sense, in contrast to p-value cutoffs.

A Monte Carlo study of Error Rates

We use a Monte Carlo study to illustrate the concepts of multiple testing adjustment.

Suppose that we have m = 300 normal populations each with variance one. Most of them have mean zero, with only three have mean one. That is, m_0=300-3=297. We test the null hypotheses of mean zero for each of the population, based on random samples of size n = 20.

We run 1,000 simulations runs (each run generates 300 samples of size n = 20 one from every population). We reject the hypotheses with four different rules:

(a) Reject those with p-value < 0.05.
(b) Reject those with Bonferroni corrected p-value < 0.05.
(c) Reject those with q-value < 0.05.
(d) Reject those with q-value < 0.4.

The R code and outputs are in the next page. We can observe the following results. Rule (a) almost always find all 3 non-zero means, but on average also find 15 more false positives (true mean zero declared as non-zero by the tests).

Rule (b) The Bonferroni correction ensure FWER of about 0.05. That is, only 5% data sets have any false positives. The price to pay is that on average we only discover 1.4 of the 3 non-zero means.

Rule (c) on average discovers 1.7 of the 3 non-zero means with only 0.15 false positives. The FWER for rule (c) is 0.13 larger than that for rule (b). The main point here is that FWER is not something we should care much. We can relax it, but control FDR and we still gets very few false positives.

Rule (d) on average discovers 2.7 of the 3 non-zero means with 2.8 false positives. Compare to (a), rule (d) recovers almost all the true positives without being overwhelmed by false positives.

This example shows that FDR (q-value) is a more appropriate quantity to focus on when we explore the data for multiple hypothesis. Particularly, the cutoff for the q-value can be adaptively changed. In this case, very few positive rejections occur at q-value < 0.05. Then we can increase the cutoff to 0.4 to recover more true positives without getting too many false positives. On the other hand, if there are already many positive rejections at q-value < 0.05, then there is no need to increase

the cutoff.

R Code

The following R code does the Monte Carlo simulation in the previous page. On the 1000 simulated data sets, the four rules (a)-(d) are applied. The average number of correct genes and false genes are summarized for each rule. These results were discussed in last page.

```
> n<-20
> nsim<-1000
> n.hyp<-300
> x0.sim<-matrix(NA, ncol=n, nrow=n.hyp)
> p.fdr<-p.bon<-p.sim<-matrix(NA, nrow=nsim, ncol=n.hyp)
> n.true<-3
> n.fdisc<-n.disc<-rep(NA,nsim)
> for (i in 1:nsim) {
+      x0.sim[,]<-rnorm(n*n.hyp)
+      x0.sim[1:n.true,] <- x0.sim[1:n.true,] + 1
+      p.sim[i,]<-apply(x0.sim,1,function(x) t.test(x,mu=0)$p.value)
+      p.bon[i,]<-p.adjust(p.sim[i,],method='bonferroni')
+      p.fdr[i,]<-p.adjust(p.sim[i,],method='fdr')
+ }
> n.disc<-apply(p.sim,1,function(x) sum(x<0.05))
> n.fdisc<-apply(p.sim[,-(1:n.true)],1,function(x) sum(x<0.05))
> mean(n.fdisc)              # average number of false positives
[1] 14.728
> mean(n.disc-n.fdisc) # average number of true positives
[1] 2.964
> fdr.tmp<-n.fdisc/n.disc
> fdr.tmp[n.disc==0]<-0# 0/0=0 in FDR definition
> mean(fdr.tmp)              # FDR
[1] 0.8244919
> mean(n.fdisc>0)            # FWER
[1] 1
>
> n.disc<-apply(p.bon,1,function(x) sum(x<0.05))
> n.fdisc<-apply(p.bon[,-(1:n.true)],1,function(x) sum(x<0.05))
> mean(n.fdisc)              # average number of false positives
[1] 0.046
> mean(n.disc-n.fdisc) # average number of true positives
[1] 1.38
> fdr.tmp<-n.fdisc/n.disc
```

```
> fdr.tmp[n.disc==0]<-0# 0/0=0 in FDR definition
> mean(fdr.tmp)               # FDR
[1] 0.02075
> mean(n.fdisc>0)             # FWER
[1] 0.044
>
> n.disc<-apply(p.fdr,1,function(x) sum(x<0.05))
> n.fdisc<-apply(p.fdr[,-(1:n.true)],1,function(x) sum(x<0.05))
> mean(n.fdisc)               # average number of false positives
[1] 0.15
> mean(n.disc-n.fdisc) # average number of true positives
[1] 1.69
> fdr.tmp<-n.fdisc/n.disc
> fdr.tmp[n.disc==0]<-0# 0/0=0 in FDR definition
> mean(fdr.tmp)               # FDR
[1] 0.04813333
> mean(n.fdisc>0)             # FWER
[1] 0.131
>
> n.disc<-apply(p.fdr,1,function(x) sum(x<0.4))
> n.fdisc<-apply(p.fdr[,-(1:n.true)],1,function(x) sum(x<0.4))
> mean(n.fdisc)               # average number of false positives
[1] 2.78
> mean(n.disc-n.fdisc) # average number of true positives
[1] 2.71
> fdr.tmp<-n.fdisc/n.disc
> fdr.tmp[n.disc==0]<-0# 0/0=0 in FDR definition
> mean(fdr.tmp)               # FDR
[1] 0.3904114
> mean(n.fdisc>0)             # FWER
[1] 0.788
```

Summary

When testing multiple hypothesis, overall error rates can be much larger than the error rate for each individual hypothesis testing. It is better to control the false discovery rate (FDR) rather than traditional family wise error rate.

The FDR adjustment can be done with the Benjamini and Hochberg (1995) procedure. This is programmed in R function p.adjust() with option method= "BH" or "fdr".

Module Summary

We have covered one-sample and two-sample tests for means and proportions. We have taught the basic concepts and terminologies for hypothesis testing. You should be able to setup the appropriate hypotheses, and carry out the tests in R. The mean tests are done with t.test(), the proportion tests are done with prop.test() and binom.test().

We discussed hypothesis testing procedures used to calculate confidence intervals.The hypothesis testing procedures in this module can be used to calculate confidence intervals.

We also taught how to use Monte Carlo simulations to find the true significance level and power of the tests.

Finally, we taught how to control error rate for multiple hypothesis testing. Particularly, we control the FDR (q-value).

Discussion

Read the paper "Interval estimation for a binomial proportion" by Lawrence D Brown, T Tony Cai, Anirban DasGupta (2001) Statistical Science pages 101-117. Available at link

http://projecteuclid.org/download/pdf_1/euclid.ss/1009213286

Write your thoughts about the following questions.

1. What are the three confidence intervals the authors recommended to use? (See the last paragraph on page 102 to the first paragraph on page 103. Also, see the section 5. Concluding remarks.) Which interval do you feel most comfortable with, and why?

2. All the recommended confidence intervals, unlike the Ward's interval, are not symmetric around the standard point estimator $\hat{p} = \frac{X}{n}$. Are the center of these confidence intervals smaller than $\hat{p} = \frac{X}{n}$ or bigger than $\hat{p} = \frac{X}{n}$? Should we use the center of these confidence intervals as point estimators for p, replacing the standard $\hat{p} = \frac{X}{n}$? Explain your recommendation.