One-sample Test for Location

Cheng Peng

STA200 Statistics II

Contents

1 Introduction

We have learned procedures for testing single population mean in elementary statistics (MAT121/125 at WCU) using aggregated information based on descriptive statistics such as sample mean, sample standard deviation, along with the distribution of the population (either a large sample size or a normal population). In this class, we will expand the procedures and introduce methods that are used in real-world applications by analysts. In addition to reviewing what we learned in MAT121/125 with assumptions about the distribution of the population in which the data was collected, we will also introduce non-parametric methods that do not assume population distribution (also called distribution-free methods).

2 (Raw) Data Structure

Instead of using sample descriptive statistics, we will work with raw data directly since you will deal with a raw dataset to solve a practical problem. The layout of a data set in practice has the following structure:

Y	X_1	X_2	$\cdots$	X_k
y_1	x_{11}	x_{21}	$\cdots$	x_{k1}
y_2	x_{12}	x_{22}	$\cdots$	x_{k2}
y_3	x_{13}	x_{23}	$\cdots$	x_{k3}

Y	X_1	X_2	$\cdots$	X_k
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
y_n	x_{1n}	x_{2n}	$\cdots$	x_{kn}

For example, an employer wants to see whether their pay rate is fair in terms of gender (i.e., assessing potential gaps in pay rate between male and female employees in the same role under similar conditions). The HR helped to create a dataset in the following form.

salary (Y)	gender (X_1)	role (X_2)	Yr_edu (X_3)	Yr_exp (X_4)
$56,000	F	analyst	12	4
$73,000	M	manager	16	8
$47,000	M	analyst	14	3
$111,000	F	manager	18	12

We also introduced R data frames earlier to store a dataset in R and then use it for analysis. The next R code defines a dataframe in R based on the above toy dataset.

```
my.toy.data <- data.frame(salary = c(56000, 73000, 47000, 111000),    # numerical values
                          gender = c("F", "M", "M", "F"),             # character (categorical) values
                          role = c("analyst", "manager", "analyst", "manager"),
                          Yr.edu = c(12, 16, 14, 18),
                          Yr.exp = c(4,8,3,12)
                          )
my.toy.data     # print the data
```

```
|   salary gender    role Yr.edu Yr.exp
| 1  56000      F analyst     12      4
| 2  73000      M manager     16      8
| 3  47000      M analyst     14      3
| 4 111000      F manager     18     12
```

If this data set is stored in a local drive or remote website in CSV (comma-separated values), we can R function `read.csv("path-to-the-file")`. See the examples from the note entitled **Getting Started with R and RStudio** in week #1.

To extract a variable from a given R data set, we use the command `dataset.name$variable`. **Caution**: R is case sensitive. For example, if we want to find the mean and variance of `salary` in the above dataframe, we can use the following code:

```
salary <- my.toy.data$salary    # extract salary from the data set with name: my.toy.data
## CAUTION: my.toy.data$Salary, my.toy.data$sAlaRy, etc. will produce errors!!!!!
average.salary <- mean(salary)  # calculate the mean salary
variance.salary <- var(salary)  # calculate variance of salary
## combined the mean and variance and print them out
cbind(average.salary = average.salary, variance.salary = variance.salary)
```

```
|      average.salary variance.salary
| [1,]          71750       800916667
```

3 One-Sample t-Test Revisited

Recall that there are different types of one-sample tests for population mean μ. We covered normal and t tests in elementary statistics. This review section focuses only on one sample t-test.

The basic assumption of the one-sample t-test is

- The population is normally distributed, say $N(\mu, \sigma)$ with σ being **unknown** population standard deviation.
- The data values are indpendently taken from the same population $N(\mu, \sigma)$. (**Note: small sample size is not an assumption for t-test!**)

Let $\{y_1, y_2, \cdots, y_n\}$ is taken independently from $N(\mu, \sigma)$ with an unknown standard deviation σ. Let $\bar{y}$ and s be the mean and standard deviation of the sample. The two-tailed one-sample t-test:

$$H_0 : \mu = \mu_0 \text{ versus } H_a : \mu \neq \mu_0$$

The test statistic for testing the above hypothesis is

$$TS = \frac{\bar{y} - \mu_0}{s/\sqrt{n}} \to t_{n-1}$$

The p-value is $P(t_{n-1} \geq |TS|)$.

We next use the well-known **Pima Indians Diabetes Dataset** at https://pengdsci.github.io/STA200/dataset/PimaIndiaDiabetes.csv. It is a commonly used dataset in statistics and machine learning, containing health data from Pima Indian women (a population with high diabetes prevalence). It is often used for binary classification (predicting diabetes) but can also be adapted for regression tasks (e.g., predicting glucose levels).

Variables in the data set are summarized in the following table.

Column Name	Description	Data Type
`pregnant`	Number of pregnancies	Integer
`glucose`	Plasma glucose concentration (2-hour OGTT)	Numeric
`pressure`	Diastolic blood pressure (mm Hg)	Numeric
`triceps`	Triceps skinfold thickness (mm)	Numeric
`insulin`	2-Hour serum insulin (mu U/ml)	Numeric
`bmi`	Body mass index (weight in kg / height^2 in m)	Numeric
`pedigree`	Diabetes pedigree function (genetic risk)	Numeric
`age`	Age (years)	Integer
`diabetes`	Outcome (1 = diabetic, 0 = non-diabetic)	Binary

According to the *National Heart, Lung, and Blood Institute*, a population is considered obese if the average body mass index (BMI) is above 30. As an illustrative example, we use the t-test on $\mu = 30$ with the assumption that the distribution of BMI is normally distributed.

$$H_0 : \mu = 30 \text{ v.s } H_a : \mu \neq 30.$$

3.1 Manual Calculation Using R

We first use manual calculation using R to translate the above formula. The following code illustrates data loading and steps for testing the above test.

```
# read data into R
PimaIndiaDiabetes <- read.csv("https://pengdsci.github.io/STA200/dataset/PimaIndiaDiabetes.csv")
##  extract BMI under the name of mass in the data set, we will rename it as BMI in the following
BMI <- PimaIndiaDiabetes$mass
## Find the sample size using the R function length()
n <- length(BMI)
xbar <- mean(BMI)    # sample mean
s <- sd(BMI)         # sample standard deviation
TS <- (xbar -30)/(s/sqrt(n))   # evaluate the test statistics
abs.TS <- abs(TS)              # absolute value of the test statistic
## Use t distribution with n-1 degrees of freedom
## The p-value of a two-tailed test is 2 times the smaller tail area
pvalue <- 2* pt(abs.TS, df = n-1, lower.tail = FALSE)     # absolute is positive,
                                                           # upper tail area is smaller.
## combine xbar, TS, df, and p-value using R function cbind() and print them out
cbind(xbar = xbar, TS = TS, d.f. = n-1, p.value = pvalue )  # print out the p-value
```

```
|          xbar       TS d.f.      p.value
| [1,] 33.08622 8.694805  391 9.708976e-17
```

The p-value is close to 0, the null hypothesis $H_0 : \mu = 30$ is rejected!

3.2 Using R Built-in Function

We can use the R built-in function `t.test()` from the R base package **{stats}** to perform the one-sample test. The syntax is

```
t.test(x,                 # variable name
       y = NULL,          # 2nd variable IF perform a two-sample t test, ignore this for a one-sample t
       alternative = c("two.sided", "less", "greater"),  # choose one of the alternative hypotheses
       mu = mu0,          # claimed value in the null hypothesis Ho
       paired = FALSE,    # is it a paired sample t test
       var.equal = FALSE, # for two sample test
       conf.level = 0.95  # confidence level for the returned confidence interval of the mean
                          # The default confidence level is 95%; you can change it to other levels
       )
```

We will use this built-in function for a two-sample test in subsequent modules. In a one-sample t test, we only need to specify `x`, `alternative`, and `mu`, and leave other arguments at their default settings. The following simple code reproduces the above result and an additional confidence interval at a 95% confidence level.

```
## use the variable of BMI defined earlier
t.test(x = BMI,
       alternative = "two.sided",
       mu = 30,
       conf.level = 0.95)
```

```
|
|   One Sample t-test
|
| data:  BMI
| t = 8.6948, df = 391, p-value < 2.2e-16
| alternative hypothesis: true mean is not equal to 30
| 95 percent confidence interval:
|  32.38837 33.78407
| sample estimates:
| mean of x
|  33.08622
```

The results above match the manual calculation exactly, as expected.

4 Regression Approach to One-sample t Tests

Recall that the simple (least) linear regression model has the following form.

$$y = \beta_0 + \beta_1 x + \epsilon,$$

The residual error ϵ is assumed to be normally distributed with a mean of 0 and an unknown standard deviation. There are two primary practical applications of linear regression:

- **Correlation Analysis**: This examines how the **mean of** y is influenced by x.
- **Predictive Analysis**: Once the intercept (β_0) and slope (β_1) are estimated, the model can be used to predict the **mean of** y for **new values of** x.

As previously discussed, the slope β_1 reflects the correlation between x and y:

- If $\beta_1 = 0$, there is no linear association between x and y.
- If $\beta_1 \neq 0$, x and y are linearly correlated, with **the sign of** β_1 indicating the direction (positive/negative) of the relationship.

Interpretation of Intercept: *The intercept (β_0) represents the expected value of y when $X = 0$ or $\beta_1 = 0$.*

For one-sample t test: There is no independent variable x that influences Y. This implies that fitting $y = \beta_0 + \epsilon$ is equivalent to eastimating β_0 as the mean of Y

In other words, one one-sample hypothesis test

$$H_0 : \mu = \mu_0 \ \text{ v.s } \ H_a : \mu \neq \mu_0$$

equivalent to testing the intercept in regression $y = \beta_0 + \epsilon$, under the above H_0,

$$H_0 : \beta_0 = \mu_0 \ \text{ v.s } \ H_a : \beta_0 \neq \mu_0,$$

Which is equivalent to

$$H_0 : \beta_0 - \mu_0 = 0 \ \text{ v.s } \ H_a : \beta_0 - \mu_0 \neq 0.$$

Using the expression in the above H_0, we re-express the constant regression model $y = \beta_0 + \epsilon$ as

$$\underbrace{y - \mu_0}_{Y'} = \underbrace{(\beta_0 - \mu_0)}_{\alpha_0} + \epsilon.$$

Inference on $H_0 : \beta_0 - \mu_0 = \alpha_0 = 0$ in the above special regression is **explicitly** given in the output in R. To implement in R, we also need to express the original y into $y - \mu_0$.

Setting up data for R regression using`lm()`:

- The model formula of the linear regression model with intercept in R is given by `lm(y.variable ~ 1, data = dataset_name)`
- **model formula accepts only variables rather than expressions in the response**. This means we cannot use the expression $y - \mu_0$ directly in the model formula. Two methods to overcome this constraint in R
  - Introduce a new variable $Z = y - \mu_0$ and use this new variable Z in the model formula with intercept only, i.e., `lm(Z~1, data = dataset_name)`.
  - A simple alternative is to use R's identity function `I()` to keep it as is (but perform an internal implicit evaluation). That is, we simply use `I(y-\mu_0)` in the formula. The complete model formula is `lm(I(y-mu0) ~ 1, data = dataset_name)`.

Next, we reproduce the above t-test on BNI in the example discussed in the last section.

```r
## If you have restarted the R session, you need to reload the data in order to proceed.
## Otherwise, simply use the data you previously loaded.
##
## claimed mean in Ho:  mu0 = 30 (mean of the BMI)
one.sample.t.reg <- lm(I(BMI-30) ~ 1, data = PimaIndiaDiabetes)
summary(one.sample.t.reg)
```

```
|
| Call:
| lm(formula = I(BMI - 30) ~ 1, data = PimaIndiaDiabetes)
|
| Residuals:
|     Min      1Q  Median      3Q     Max
| -14.886  -4.686   0.114   4.014  34.014
|
| Coefficients:
|             Estimate Std. Error t value Pr(>|t|)
| (Intercept)    3.086      0.355   8.695   <2e-16 ***
| ---
| Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
|
| Residual standard error: 7.028 on 391 degrees of freedom
```

<font color = "red" size 4>**Interpretation of Output:**

- $\overline{y - \mu_0} = \bar{Z} = 3.086$
- The test statistic

$$TS = \frac{\overline{y-\mu_0} - 0}{se(\overline{y-\mu_0})} \to t_{n-1}$$

  that is,

$$TS = \frac{3.086 - 0}{0.355} = 8.695$$

- The pvalue is `2*qt(8.695, df = 391) < 2e-16` which is approximately equal to 0.
- $\overline{y - \mu_0} = 3.086$ is equivalent to $\bar{y} - 30 = 3.086$ which gives $\bar{y} = 30 + 3.086 = 33.086$

The above explanation shows that the regression approach reproduces the same results as the methods covered in MAT121/125. This means that the one-sample t-test is indeed a degenerate case of linear regression where the only parameter is the intercept. This connection shows that many statistical tests are unified under the **general linear model framework**.

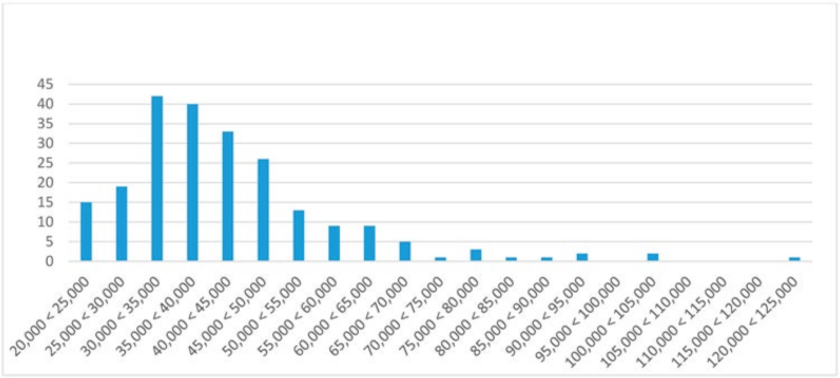
5 One-sample Sign Test

The one-sample t-test for **population mean** (μ) assumes that the dataset was drawn from a normally distributed population. This is a strong and crucial assumption for the t-test and normal linear regression models, particularly in the case of small samples.

However, it is not uncommon to have distributions in real-world applications that are normally distributed. For example, starting salaries of graduates from a university are **not** normally distributed.

Number of Employees by Salary Range Distribution

Source: W Washington Daily News



We also learned from introductory statistics that the **mean**, as a measure of central tendency, is sensitive to the shape of a distribution. The **median**, on the other hand, is a more robust measure of the center for numerical populations. Therefore, we use **media** to measure the density of populations. Inference on the center should also be based on **median** instead of **mean**.

In this section, we introduce a **nonparametric sign test** for the population **median**, which is particularly useful for skewed distributions. The term **nonparametric test** means that **the test does not rely on any distributional assumptions**.

CAUTION:

Non-parametric procedures (also called distribution-free procedures) do not assume any parametric distribution about populations. However, the sampling distribution of the test derived based on the sample taken from the population has a parametric distribution (or can be approximated by a parametric distribution!). The large sample normal test for population mean is an example.

The one-sample sign test is a nonparametric statistical method used to determine whether the median of a single sample differs from a hypothesized value. Unlike parametric tests (e.g., the one-sample t-test), the sign test does not assume any specific distribution (e.g., normality) and is robust to outliers.

The sign test is primarily used when:

- The data are ordinal or continuous but **not normally distributed**.
- The sample size is **small**, making parametric tests unreliable.
- The data contain **outliers** that could distort mean-based tests.

Before introducing the testing procedure, we first outline the binomial distribution, which will be used to characterize the distribution of the sign test statistic.

5.1 Binomial Distribution

The binomial distribution is a discrete probability distribution that models the number of successes in a fixed number of independent trials, where each trial has only two possible outcomes: success or failure. A binomial random variable has the following characteristics.

- **Number of trials (n)**: Fixed number of experiments.
- **Probability of success (p)**: Constant for each trial.
- **Probability of failure ($\mathbf{q = 1 - p}$)**: Complement of success.
- **Independent trials**: The Outcome of one trial does not affect the others.

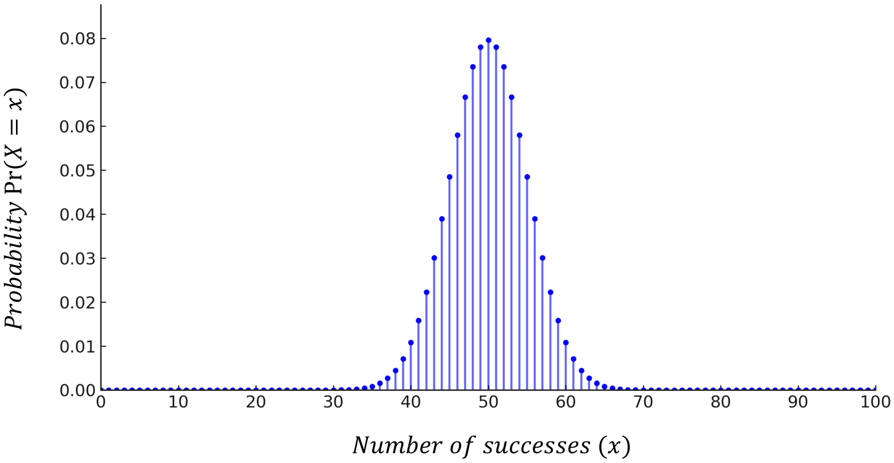
The probability distribution function is explicitly given by

$$P(X = k) = \frac{n!}{k!(n-k)!}p^k(1-p)^{n-k}$$

where

- X is the number of successes,
- k is a specific number of successes ($0 \leq k \leq n$).

The following probability distribution histogram is based on a special binomial distribution with 100 trials and a success probability of 0.5



Similar to other distributions such as normal, t, and chi-square, we need to know how to find tail probabilities (for p-values) and percentiles (critical values). Two R functions will be used:

- `pbinom(q, size, prob, lower.tail = TRUE)` for left tail probability
  - `q` - Number of successes (can be a vector).
  - `size` - Number of trials (n). +`prob` Probability of success (p).
  - `lower.tail` - Logical; if TRUE (default), returns $P(X \leq q)$; if FALSE, returns $P(X > q)$.
- `qbinom(p, size, prob, lower.tail = TRUE)` for percentile (also called *quantile*)
  - `p` - The cumulative probability (a value between 0 and 1).
  - `size` - Number of trials (n).
  - `prob` - Probability of success (p).

  - `lower.tail` - If TRUE (default), returns x such that $P(X \leq x) \geq p$; if FALSE, returns x such that $P(X > x) \leq p$.

Some textbooks include a binomial distribution table with a small n (usually not more than 20) and a few specific probabilities (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9). I also make a binomial table at

|Binomial Table [PDF]|

Example 1: What is the 90th percentile for a binomial distribution with 20 trials and success probability 0.5?

```
qbinom(0.9, size = 20, prob = 0.5)
```

```
| [1] 13
```

Example 2: What is the probability of getting more than 3 heads in 10 tosses?

```
pbinom(3, size = 10, prob = 0.5, lower.tail = FALSE)
```

```
| [1] 0.828125
```

5.2 Test Formulation

Let M be the median of the population from which the random sample is taken. The hypotheses are defined as one of the following three forms:

- Two-tailed test: $H_0 : M = M_0 \;\text{ vs }\; H_a : M \neq M_0$.
- Right-tailed test: $H_0 : M \leq M_0 \;\text{ vs }\; H_a : M > M_0$.
- Left-tailed test: $H_0 : M \geq M_0 \;\text{ vs }\; H_a : M < M_0$.

The calculation of the test statistic is based on the following steps:

Step 1: Calculate the differences between each observed value and the hypothesized median M_0.

$$D_i = Y_i - M_0$$

Step 2: Count signs of the differences obtained in **Step 1**

- Positive signs: Number of observations $> M_0$.
- Negative signs: Number of observations $< M_0$.
- Zeros: Exclude all zeros (if any).

Step 3: Calculate the test statistic. Let n be the number of **non-zero** differences (**i.e., the sum of both positive and negative signs**). Under the null hypothesis H_0, the test statistic defined below follows a binomial distribution.

$$S = \text{number of positive (or negative) signs} \to \text{Binom}(n, p = 0.5).$$

Step 3.1: If $n \geq 25$, the following standardized statistic can be approximated to the standard normal distribution.

$$Z = \frac{S - 0.5 \times n}{0.5\sqrt{n}} \to N(0, 1)$$

Therefore, we can use a normal table to find the p-value for testing the hypothesis.

5.3 An Illustrative Example

We use a toy example to illustrate the above sign test procedure through manual calculation (R is used as a calculator).

Data: `230, 245, 252, 248, 255, 260, 235, 240, 250, 242, 238, 258, 245, 248, 253`

```
# Data
reaction.time <- c(230, 245, 252, 248, 255, 260, 235, 240, 250, 242, 238, 258, 245, 248, 253)
```

HYpothesis: $H_0 : M = 250$ vs $H_a : M \neq 250$ (two-tailed test)

Steps: Taken difference - $X_i - 250$

```
reaction.time - 250
```

```
|  [1] -20  -5   2  -2   5  10 -15 -10   0  -8 -12   8  -5  -2   3
```

- Positive Signs

```
pos.sign <- sum((reaction.time - 250 )>0)
pos.sign
```

```
| [1] 5
```

- Negative Signs

```
neg.sign <- sum((reaction.time - 250) < 0)
neg.sign
```

```
| [1] 9
```

- Zeros (excluded)

```
zeros <- sum((reaction.time - 250) == 0)
zeros
```

```
| [1] 1
```

- Test Statistic $TS = 5$ or $TS = 9$ and $n = 14$.
- pvalue $P(S \geq 9) = 0.395)$

```
pval = 2*pbinom(9, size = 14, p = 0.5, lower.tail = FALSE)
pval
```

```
| [1] 0.1795654
```

5.4 Implementation with R

Two R functions, `binom.test()` in package **{stats}** and `SIGN.test` in package **{BSDA}**, can be used to perform a sign test for the population median.

`binom.test()` requires providing descriptive statistics: number of positive signs (TS), number of trials (n = sum of positive and negative signs).

```
# use the descriptive statistics calculated earlier
binom.test(pos.sign, n = (pos.sign+neg.sign), p = 0.5, alternative = "two.sided" )
```

```
|
|   Exact binomial test
|
| data:  pos.sign and (pos.sign + neg.sign)
| number of successes = 5, number of trials = 14, p-value = 0.424
| alternative hypothesis: true probability of success is not equal to 0.5
| 95 percent confidence interval:
|  0.1275984 0.6486199
| sample estimates:
| probability of success
|              0.3571429
```

Explanation of the output:

- n = 14.
- TS = 5 (or 15-4 = 9)
- p.value = 0.424.
- 95% confidenc interval of the **true percentage of negative signs**: (0.1275984, 0.6486199).
- Probability of success: $ number of negative signs/n = 4/15 = 0.3571429$.

CAUTION: *The output reports inferential statistics that depend on the proportion of negative signs defined based on the difference between the original data values and the hypothesized median. The confidence interval is NOT for the median of the original population!*

`SIGN.test` performs the test directly based on the data and the hypothesized median (in the null hypothesis H_0), since the package **BSDA** is not included in the base R. We need to install and load it before calling the function `SIGN.test()`

```
# install and load the package
if (!require("BSDA")) {
   install.packages("BSDA")
   library(BSDA)
}
# Call SIGN.test() to perform the sign test
SIGN.test(reaction.time, md = 250)
```

```
|
|   One-sample Sign-Test
|
| data:  reaction.time
| s = 5, p-value = 0.424
| alternative hypothesis: true median is not equal to 250
| 95 percent confidence interval:
|  240.3563 252.8218
| sample estimates:
| median of x
|         248
|
| Achieved and Interpolated Confidence Intervals:
|
|                   Conf.Level   L.E.pt   U.E.pt
| Lower Achieved CI     0.8815 242.0000 252.0000
| Interpolated CI       0.9500 240.3563 252.8218
| Upper Achieved CI     0.9648 240.0000 253.0000
```

It is not supprised that the test statistic and p-value are identical to those produced in the output of `binom.test()` and manual calculation. `SIGN.test()` reports some inferential statistics that are based on the original scale of the population:

- Estimated median: $\bar{M} = 248$.

- 95% confidence interval for the population median is (240.3563, 252.8218)

Please ignore the reported confidence intervals for the lower and upper achieved median!

The following short video (https://pengdsci.github.io/STA200/week03/Nonparamtrics-SignedTest.mp4) explains the sign test with a manually worked example. **Table 8** in the video refers to the binomial distribution table. You can find the link to this table in the **first subsection above**.

5.5 Concluding Remark

We discussed how to regression approach to the one-sample t test. The question is whether there is a similar regression approach to testing the population median. The answer is yes. Linear regression models build a functional relationship between the mean population and factors that influence the population mean. There are also regression models to characterize the relationship between quantiles of a population and factors that influence the population quantiles - quantile regression. Median is a special quantile; this means, we can also use **quantile regression** to test the population median. The method of quantile regression is out of the scope of this course.

The one-sample sign test is a simple, distribution-free method for testing medians. While it is robust and easy to apply, it sacrifices some statistical power compared to parametric alternatives. It remains useful for skewed data, small samples, or when outliers are a concern.