Probability Distributions and Statistical Inference in R

Instructor: Dr. Azadeh Alimadad
Department of Statistics & Data Analytics

Course Overview

This lecture introduces probability distributions and statistical inference using R. We'll cover discrete and continuous distributions, hypothesis testing and confidence intervals. Practical R implementations will be demonstrated throughout.

Contents

1 Introduction to Probability Distributions in R

1.1 R's Distribution Functions

R uses a consistent naming convention for probability distributions:

- **d**: Density/mass function (density)
- **p**: Cumulative distribution function (probability)
- **q**: Quantile function (quantile)
- **r**: Random number generation (random)

1.2 Common Distribution Functions

```
# General syntax
dxxxx(x, parameters)    \# Density/Probability at x
pxxxx(q, parameters)    \# P(X $\leq$ q)
qxxxx(p, parameters)    \# Quantile for probability p
rxxxx(n, parameters)    \# Generate n random values
```

2 Discrete Probability Distributions

2.1 Binomial Distribution

2.1.1 Theory and Formula

The binomial distribution models the number of successes in n independent trials, each with probability of success p.

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, 1, 2, \ldots, n \tag{1}$$

Where:

- n: number of trials
- p: probability of success
- k: number of successes

2.1.2 Mean and Variance

$$E[X] = np \tag{2}$$
$$Var[X] = np(1-p) \tag{3}$$

2.1.3 R Implementation

```r
# Binomial distribution functions
n <- 10
p <- 0.3

# Probability mass function
dbinom(3, size = n, prob = p)  # P(X = 3)
dbinom(0:10, size = n, prob = p)  # All probabilities

# Cumulative distribution function
pbinom(4, size = n, prob = p)  # P(X $\leq$ 4)
pbinom(4, size = n, prob = p, lower.tail = FALSE)  # P(X $>$ 4)

# Quantile function
qbinom(0.95, size = n, prob = p)  # 95th percentile

# Random number generation
rbinom(100, size = n, prob = p)  # 100 random binomial values

# Plot binomial distribution
x <- 0:n
prob <- dbinom(x, size = n, prob = p)
plot(x, prob, type = "h", lwd = 2,
     main = "Binomial Distribution (n=10, p=0.3)",
     xlab = "Number of Successes",
     ylab = "Probability",
     col = "blue")
points(x, prob, pch = 16, col = "red")
```

2.2 Poisson Distribution

2.2.1 Theory and Formula

The Poisson distribution models the number of events occurring in a fixed interval of time or space.

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \ldots \tag{4}$$

Where λ is the average rate of occurrence.

2.2.2 Mean and Variance

$$E[X] = \lambda \tag{5}$$
$$Var[X] = \lambda \tag{6}$$

2.2.3 R Implementation

```r
# Poisson distribution functions
lambda <- 3.5

# Probability mass function
dpois(2, lambda = lambda)       # P(X = 2)
dpois(0:10, lambda = lambda)    # First 11 probabilities

# Cumulative distribution function
ppois(5, lambda = lambda)       # P(X $\leq$ 5)

# Quantile function
qpois(0.9, lambda = lambda)     # 90th percentile

# Random number generation
rpois(100, lambda = lambda)     # 100 random Poisson values

# Plot Poisson distribution
x <- 0:15
prob <- dpois(x, lambda = lambda)
plot(x, prob, type = "h", lwd = 2,
     main = paste("Poisson Distribution ($\lambda$ =", lambda, ")
        "),
     xlab = "Number of Events",
     ylab = "Probability",
     col = "darkgreen")
points(x, prob, pch = 16, col = "red")

# Example: Customer arrivals
# Suppose customers arrive at average rate of 4.2 per hour
arrivals <- rpois(100, lambda = 4.2)
hist(arrivals, breaks = seq(-0.5, max(arrivals)+0.5, 1),
     main = "Customer Arrivals per Hour",
     xlab = "Number of Customers",
     ylab = "Frequency",
     col = "lightblue")
```

3 Continuous Probability Distributions

3.1 Normal Distribution

3.1.1 Theory and Formula

The probability density function of the normal distribution is:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}, \quad -\infty < x < \infty \tag{7}$$

Where:

- μ: mean

- σ: standard deviation

3.1.2 Standard Normal Distribution

When $\mu = 0$ and $\sigma = 1$, we have the standard normal distribution $Z \sim N(0, 1)$.

3.1.3 R Implementation

```r
# Normal distribution functions
mu <- 100
sigma <- 15

# Density function
dnorm(110, mean = mu, sd = sigma)         # f(110)
dnorm(seq(70, 130, by = 5), mu, sigma)    # Multiple values

# Cumulative distribution function
pnorm(110, mean = mu, sd = sigma)         # P(X $\leq$ 110)
pnorm(110, mean = mu, sd = sigma, lower.tail = FALSE)  # P(X $>$
    110)

# Standard normal
pnorm(1.96)                                # P(Z $\leq$ 1.96) =
    0.975
pnorm(-1.96)                               # P(Z $\leq$ -1.96) =
    0.025

# Quantile function
qnorm(0.95, mean = mu, sd = sigma)        # 95th percentile
qnorm(0.975)                               # z-score for 95% CI:
    1.96
qnorm(c(0.025, 0.975))                     # Critical values for
    95% CI

# Random number generation
rnorm(100, mean = mu, sd = sigma)         # 100 random values

# Plot normal distribution
x <- seq(50, 150, length.out = 200)
pdf <- dnorm(x, mean = mu, sd = sigma)
plot(x, pdf, type = "l", lwd = 2,
     main = "Normal Distribution ($\mu$=100, $\sigma$=15)",
     xlab = "x",
     ylab = "Density",
     col = "blue")

# Shade areas
# P(85 $\leq$ X $\leq$ 115)
x_shade <- seq(85, 115, length.out = 100)
y_shade <- dnorm(x_shade, mean = mu, sd = sigma)
polygon(c(85, x_shade, 115), c(0, y_shade, 0),
```

```
        col = rgb(0, 0, 1, 0.3), border = NA)
```

3.2 t-Distribution

3.2.1 Theory and Formula

The t-distribution is used for small sample sizes when population variance is unknown.

$$f(t) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}} \tag{8}$$

Where ν is degrees of freedom.

3.2.2 R Implementation

```r
# t-distribution functions
df <- 10  # degrees of freedom

# Density function
dt(1.5, df = df)                # f(1.5)

# Cumulative distribution function
pt(2.0, df = df)                # P(t $\leq$ 2.0)
pt(2.0, df = df, lower.tail = FALSE)  # P(t $>$ 2.0)

# Critical values for confidence intervals
qt(0.975, df = df)              # t-value for 95% CI
qt(0.995, df = 20)              # t-value for 99% CI with df=20

# Compare t-distribution with normal distribution
x <- seq(-4, 4, length.out = 200)
plot(x, dnorm(x), type = "l", lwd = 2, col = "black",
     main = "t-Distribution vs Normal Distribution",
     xlab = "x", ylab = "Density")
lines(x, dt(x, df = 1), col = "red", lwd = 2)
lines(x, dt(x, df = 5), col = "blue", lwd = 2)
lines(x, dt(x, df = 30), col = "green", lwd = 2)
legend("topright",
       legend = c("Normal", "t (df=1)", "t (df=5)", "t (df=30)"),
       col = c("black", "red", "blue", "green"), lwd = 2)

# t-test example
set.seed(123)
sample_data <- rnorm(15, mean = 105, sd = 10)
t_test_result <- t.test(sample_data, mu = 100)
print(t_test_result)
cat("t-statistic:", t_test_result$statistic, "\n")
cat("p-value:", t_test_result$p.value, "\n")
cat("95% CI:", t_test_result$conf.int, "\n")
```

4 Chi-Square Distribution

4.1 Theory and Formula

The chi-square distribution with k degrees of freedom is the sum of squares of k independent standard normal variables.

$$f(x;k) = \frac{1}{2^{k/2}\Gamma(k/2)} x^{k/2-1} e^{-x/2}, \quad x > 0 \tag{9}$$

4.2 Applications

1. Confidence interval for variance
2. Goodness-of-fit tests
3. Test of independence
4. Test of homogeneity

4.3 R Implementation

```r
# Chi-square distribution functions
df <- 5  # degrees of freedom

# Density function
dchisq(3, df = df)              # f(3)

# Cumulative distribution function
pchisq(10, df = df)             # P($\chi^2 \leq$ 10)

# Critical values
qchisq(0.95, df = df)           # 95th percentile
qchisq(c(0.025, 0.975), df = df)  # Critical values for 95% CI

# Plot chi-square distributions
x <- seq(0, 20, length.out = 200)
plot(x, dchisq(x, df = 1), type = "l", lwd = 2, col = "red",
     main = "Chi-Square Distributions",
     xlab = "x", ylab = "Density",
     ylim = c(0, 0.5))
lines(x, dchisq(x, df = 3), col = "blue", lwd = 2)
lines(x, dchisq(x, df = 5), col = "green", lwd = 2)
lines(x, dchisq(x, df = 10), col = "orange", lwd = 2)
legend("topright",
       legend = c("df=1", "df=3", "df=5", "df=10"),
       col = c("red", "blue", "green", "orange"), lwd = 2)
```

4.4 Confidence Interval for Variance

4.4.1 Theory

For a sample from a normal distribution:

$$\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1} \tag{10}$$

The $(1-\alpha)$ confidence interval for σ^2 is:

$$\left[\frac{(n-1)s^2}{\chi^2_{1-\alpha/2,n-1}}, \frac{(n-1)s^2}{\chi^2_{\alpha/2,n-1}}\right] \tag{11}$$

4.4.2 R Implementation

```r
# Function for confidence interval of variance
ci_variance <- function(data, alpha = 0.05) {
  n <- length(data)
  s2 <- var(data)
  df <- n - 1

  # Chi-square critical values
  chi_lower <- qchisq(alpha/2, df, lower.tail = FALSE)
  chi_upper <- qchisq(1 - alpha/2, df, lower.tail = FALSE)

  # Confidence interval for variance
  ci_var <- c((n-1)*s2/chi_lower, (n-1)*s2/chi_upper)

  # Confidence interval for standard deviation
  ci_sd <- sqrt(ci_var)

  return(list(variance = s2,
              sd = sd(data),
              ci_variance = ci_var,
              ci_sd = ci_sd))
}

# Example
set.seed(123)
sample_data <- rnorm(25, mean = 100, sd = 15)
result <- ci_variance(sample_data, alpha = 0.05)

cat("Sample variance:", result$variance, "\n")
cat("Sample standard deviation:", result$sd, "\n")
cat("95% CI for variance:", result$ci_variance, "\n")
cat("95% CI for standard deviation:", result$ci_sd, "\n")

# Verify with built-in function
# install.packages("DescTools")
library(DescTools)
VarTest(sample_data)
```

4.5 Goodness-of-Fit Test

4.5.1 Theory

Tests whether observed frequencies match expected frequencies:

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i} \sim \chi^2_{k-1} \tag{12}$$

4.5.2 R Implementation

```r
# Goodness-of-fit test example

# Example 1: Dice fairness test
observed <- c(22, 18, 15, 19, 24, 22)  # Observed frequencies for 6 faces
expected <- rep(20, 6)                  # Expected frequencies if fair

chi_test <- chisq.test(observed, p = rep(1/6, 6))
print(chi_test)

# Extract components
cat("Chi-square statistic:", chi_test$statistic, "\n")
cat("p-value:", chi_test$p.value, "\n")
cat("Degrees of freedom:", chi_test$parameter, "\n")

# Example 2: Mendelian genetics
# 9:3:3:1 ratio expected
observed_plants <- c(315, 101, 108, 32)  # Observed counts
expected_ratio <- c(9/16, 3/16, 3/16, 1/16)  # Expected ratios

chi_plants <- chisq.test(observed_plants, p = expected_ratio)
print(chi_plants)

# Manual calculation
total <- sum(observed_plants)
expected_counts <- total * expected_ratio
chi_sq <- sum((observed_plants - expected_counts)^2 / expected_counts)
p_value <- pchisq(chi_sq, df = 3, lower.tail = FALSE)

cat("\nManual calculation:\n")
cat("Chi-square:", chi_sq, "\n")
cat("p-value:", p_value, "\n")
```

5 F-Distribution and ANOVA

5.1 Theory and Formula

The F-distribution arises as the ratio of two independent chi-square variables divided by their degrees of freedom.

$$F = \frac{U_1/d_1}{U_2/d_2} \sim F(d_1, d_2) \tag{13}$$

Where $U_1 \sim \chi^2_{d_1}$ and $U_2 \sim \chi^2_{d_2}$.

5.2 R Implementation

```r
# F-distribution functions
df1 <- 3   # numerator degrees of freedom
df2 <- 20  # denominator degrees of freedom

# Density function
df(2.5, df1 = df1, df2 = df2)      # f(2.5)

# Cumulative distribution function
pf(3.0, df1 = df1, df2 = df2)      # P(F $\leq$ 3.0)

# Critical values
qf(0.95, df1 = df1, df2 = df2)     # 95th percentile
qf(0.99, df1 = 2, df2 = 15)        # Critical value for $\alpha$ =
    0.01

# Plot F-distributions
x <- seq(0, 5, length.out = 200)
plot(x, df(x, df1 = 1, df2 = 10), type = "l", lwd = 2, col = "red
    ",
     main = "F-Distributions",
     xlab = "x", ylab = "Density",
     ylim = c(0, 1))
lines(x, df(x, df1 = 3, df2 = 10), col = "blue", lwd = 2)
lines(x, df(x, df1 = 5, df2 = 10), col = "green", lwd = 2)
lines(x, df(x, df1 = 10, df2 = 10), col = "orange", lwd = 2)
legend("topright",
       legend = c("F(1,10)", "F(3,10)", "F(5,10)", "F(10,10)"),
       col = c("red", "blue", "green", "orange"), lwd = 2)
```

5.3 One-Way ANOVA

5.3.1 Theory

Tests whether means of several groups are equal:

$$SS_{total} = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \bar{x})^2 \tag{14}$$

$$SS_{between} = \sum_{i=1}^{k} n_i(\bar{x}_i - \bar{x})^2 \tag{15}$$

$$SS_{within} = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \bar{x}_i)^2 \tag{16}$$

$$F = \frac{MS_{between}}{MS_{within}} = \frac{SS_{between}/(k-1)}{SS_{within}/(N-k)} \sim F(k-1, N-k) \tag{17}$$

5.3.2 R Implementation

```r
# One-way ANOVA example

# Create sample data
set.seed(123)
group1 <- rnorm(20, mean = 100, sd = 10)
group2 <- rnorm(20, mean = 110, sd = 10)
group3 <- rnorm(20, mean = 105, sd = 10)
group4 <- rnorm(20, mean = 115, sd = 10)

# Combine into data frame
data_anova <- data.frame(
  value = c(group1, group2, group3, group4),
  group = factor(rep(1:4, each = 20))
)

# Perform ANOVA
anova_result <- aov(value ~ group, data = data_anova)
summary(anova_result)

# Extract F-statistic and p-value
f_stat <- summary(anova_result)[[1]]$"F value"[1]
p_value <- summary(anova_result)[[1]]$"Pr(>F)"[1]

cat("F-statistic:", f_stat, "\n")
cat("p-value:", p_value, "\n")

# Manual calculation for understanding
group_means <- tapply(data_anova$value, data_anova$group, mean)
overall_mean <- mean(data_anova$value)
n_groups <- length(unique(data_anova$group))
n_total <- nrow(data_anova)

```

```r
# Calculate sums of squares
ss_between <- sum(table(data_anova$group) * (group_means - overall_mean)^2)
ss_within <- sum(tapply(data_anova$value, data_anova$group,
                        function(x) sum((x - mean(x))^2)))
ss_total <- sum((data_anova$value - overall_mean)^2)

# Calculate mean squares
ms_between <- ss_between / (n_groups - 1)
ms_within <- ss_within / (n_total - n_groups)

# Calculate F-statistic
f_calc <- ms_between / ms_within
p_calc <- pf(f_calc, df1 = n_groups - 1, df2 = n_total - n_groups,
             lower.tail = FALSE)

cat("\nManual calculation:\n")
cat("SS_between:", ss_between, "\n")
cat("SS_within:", ss_within, "\n")
cat("MS_between:", ms_between, "\n")
cat("MS_within:", ms_within, "\n")
cat("F-statistic:", f_calc, "\n")
cat("p-value:", p_calc, "\n")

# Post-hoc test (Tukey's HSD)
if (p_value < 0.05) {
  tukey_result <- TukeyHSD(anova_result)
  print(tukey_result)
  plot(tukey_result)
}
```

6 Exercises

6.1 Basic Exercises

1. Binomial Distribution:
   - Simulate 1000 coin flips (fair coin) and plot the distribution
   - Calculate probability of getting 7 or more heads in 10 flips
   - Find the number of successes corresponding to the 95th percentile
2. Normal Distribution:
   - Generate 500 values from N(100, 15) and verify empirical vs theoretical probabilities
   - Calculate the probability that a value is between 85 and 115
   - Find the 99th percentile

3. t-Distribution:
    - Compare t-distribution with 5, 15, and 30 df to standard normal
    - Calculate critical values for 90%, 95%, and 99% confidence intervals with df=10

6.2 Intermediate Exercises

1. Chi-Square Applications:
    - Test if a 6-sided die is fair using 60 rolls with observed frequencies (9, 11, 10, 8, 12, 10)
    - Calculate 95% CI for variance of a sample from N(0,1) with n=25
2. ANOVA:
    - Simulate data for 3 groups with different means and perform ANOVA
    - Check ANOVA assumptions (normality, homogeneity of variance)
    - Perform post-hoc tests if ANOVA is significant

6.3 Advanced Exercises

1. Comprehensive Analysis:
    - Design a complete study with multiple groups and covariates
    - Perform all appropriate statistical tests
    - Create a complete report with interpretations
2. Power Analysis:
    - Calculate required sample size for detecting effect sizes in t-tests and ANOVA
    - Simulate power curves for different effect sizes

7 Resources and References

7.1 Books

- "Introduction to Probability and Statistics Using R" by G. Jay Kerns
- "The R Book" by Michael J. Crawley
- "Statistical Inference via Data Science" by Chester Ismay and Albert Y. Kim

7.2 Online Resources

- R Documentation: `https://www.rdocumentation.org/`
- Quick-R: `https://www.statmethods.net/`
- CRAN Task Views: `https://cran.r-project.org/web/views/`
- R-bloggers: `https://www.r-bloggers.com/`

7.3 Packages for Advanced Analysis

```r
# Useful packages for probability and statistics
install.packages(c(
  "ggplot2",      # Advanced plotting
  "dplyr",        # Data manipulation
  "tidyr",        # Data tidying
  "broom",        # Tidy model outputs
  "infer",        # Tidy statistical inference
  "effects",      # Effect displays
  "lsr",          # Learning statistics with R
  "pwr"           # Power analysis
))
```

Conclusion

This lecture covered essential probability distributions and statistical inference methods in R. Understanding these concepts is crucial for data analysis and interpretation. Remember to:

1. Choose appropriate distributions for your data 2. Check assumptions before applying tests 3. Interpret results in context 4. Report estimates with confidence intervals 5. Visualize your data and results

Practice is key to mastering these concepts. Start with simple examples and gradually work on more complex analyses.

Remember: Statistics is about understanding uncertainty, not eliminating it.