Some preliminaries

Advantages (HWC: Chapter 1)

1. Few assumptions on the underlying populations
2. Enable to compute exact P-values, coverage probabilities
3. Often easier to apply than their parametric counterparts
4. Easy to understand
5. Slightly less efficient than parametric test when parametric assumptions hold
6. Robust to outliers
7. Can be used in many practical situations where theory is intractable
8. Nonparametric + prior information $\rightarrow$ Bayesian nonparametric approach

There are two ways to understand a statistical method: either 1) mathematically or 2) computationally (coding). Nonparametric methods usually need both.

Notations

- X: random variable, representing a quantity of interest (e.g., a biological or physical measurement). Another way to view a random variable is as a function that maps outcomes of an experiment to real numbers. It is called random because the underlying experiment can produce different outcomes, each occurring with a certain probability, which in turn induces randomness in the value of the function.
- x: realizations (observed random variables)
- $f(x)$: probability density function (pdf)
- $F_X(x) = P(X \leq x)$: cumulative distribution function (cdf) [The cumulative distribution function (CDF) of X is defined as

  $$F_X(x) = P(X \leq x),$$

  which gives the probability that X takes a value less than or equal to x. Intuitively, $F_X(x)$ represents the proportion of the population whose values lie at or below x. As x increases, $F_X(x)$ accumulates probability and ranges from 0 to 1.] Importantly, it can only be defined for a univariate random variable.
- $X_1, \ldots, X_n$: random sample (independent and identically distributed)

Some examples

Example 1. Linear regression

Matrix–vector representation of linear regression. Let $(y_i, \boldsymbol{x}_i)$, $i = 1, \ldots, n$, be observations, where $y_i \in \mathbb{R}$ and $\boldsymbol{x}_i \in \mathbb{R}^p$ is a vector of covariates. Stacking the observations gives the linear regression model

$$\mathbf{y} = \boldsymbol{X}\boldsymbol{\beta} + \varepsilon,$$

where

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n, \qquad \boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1^\top \\ \vdots \\ \boldsymbol{x}_n^\top \end{pmatrix} \in \mathbb{R}^{n \times p}, \qquad \boldsymbol{\beta} \in \mathbb{R}^p, \qquad \boldsymbol{\varepsilon} \in \mathbb{R}^n.$$

Matrix–vector products. For a matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and a vector $\boldsymbol{\beta} \in \mathbb{R}^p$, the product $\boldsymbol{X}\boldsymbol{\beta} \in \mathbb{R}^n$ is defined componentwise by

$$(\boldsymbol{X}\boldsymbol{\beta})_i = \sum_{j=1}^{p} X_{ij}\beta_j, \qquad i = 1, \ldots, n.$$

In linear regression, $(\boldsymbol{X}\boldsymbol{\beta})_i = \boldsymbol{x}_i^\top \boldsymbol{\beta}$ is the fitted value corresponding to the ith observation.

Transpose and inner products. For vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^p$, the inner product is $\boldsymbol{u}^\top \boldsymbol{v} = \sum_{j=1}^{p} u_j v_j$. The transpose $\boldsymbol{X}^\top \in \mathbb{R}^{p \times n}$ satisfies

$$\boldsymbol{X}^\top \mathbf{y} = \begin{pmatrix} \sum_{i=1}^{n} x_{i1} y_i \\ \vdots \\ \sum_{i=1}^{n} x_{ip} y_i \end{pmatrix},$$

which collects covariate–response cross-products.

Normal equations. Thus the errors are $e_i = y_i - \boldsymbol{x}_i\boldsymbol{\beta}$ The ordinary least squares estimator minimizes

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i\boldsymbol{\beta})^2 = (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})^\top (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2,$$

We can get the least-squared estimate as (by calculating the $\boldsymbol{\beta}$ for which the sum of 'squared' errors is the 'least')

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i\boldsymbol{\beta})^2$$

This is obtained by solving the first derivative condition $\frac{\partial}{\partial \boldsymbol{\beta}} (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})^\top (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}) = 0$ to get the $\boldsymbol{\beta}$ that minimizes the sum of squared errors $(\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})^\top (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})$,

$$\boldsymbol{X}^\top \boldsymbol{X}\, \hat{\boldsymbol{\beta}} = \boldsymbol{X}^\top \mathbf{y}.$$

This is also called the 'normal' equation.

If $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible, then

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \mathbf{y}.$$

Geometric interpretation. The vector $\boldsymbol{X}\widehat{\boldsymbol{\beta}}$ is the orthogonal projection of $\mathbf{y}$ onto the column space of $\boldsymbol{X}$. The residual vector $\mathbf{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}$ is orthogonal to every column of $\boldsymbol{X}$, that is,

$$\boldsymbol{X}^\top(\mathbf{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}) = 0.$$

Nonparametric methods are not only about nonparametric models, but also a way of thinking nonparametrically. We illustrate this through three regression settings. The only thing that changes is the assumption on the error.

0.1 Case 1: Parametric model with homoscedastic normal errors

Assume the classical linear model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 I).$$

- The full distribution of ε is specified.
- Ordinary least squares (OLS) is the maximum likelihood estimator.
- Exact finite-sample inference is available.
- $\widehat{\boldsymbol{\beta}}_{\text{OLS}}$ achieves the Cramér–Rao lower bound. (It states that the precision of any unbiased estimator is at most the Fisher information; or (equivalently) the reciprocal of the Fisher information is a lower bound on its variance. An unbiased estimator that achieves this bound is said to be (fully) efficient. Such a solution achieves the lowest possible mean squared error among all unbiased methods, and is, therefore, the minimum variance unbiased (MVU) estimator. Why is this important? Since estimators are functions of the data, the data variability will always contribute to variability in the estimate. However, we would want an estimator that does not vary too much due to data variability. Hence, this lower bound will allow us to identify the estimators that achieve this bound, and those are expected to be more stable against data variability.)
- Classical t-tests are exact and efficient.

0.2 Case 2: Multivariate normal errors with unknown covariance

Assume

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma}),$$

where $\boldsymbol{\Sigma}$ is an unknown positive definite covariance matrix.

- The error distribution is Gaussian, but $\boldsymbol{\Sigma}$ is unspecified.
- OLS remains consistent but is no longer efficient.
- The asymptotic variance of $\widehat{\boldsymbol{\beta}}$ depends on $\boldsymbol{\Sigma}$.
- Exact inference is unavailable; inference is asymptotic.
- 2WLS can be used. Two-step weighted least squares (2WLS) is a feasible version of GLS:
  1. Obtain a preliminary consistent estimator (e.g., OLS) and residuals $\widehat{\boldsymbol{\varepsilon}} = \boldsymbol{Y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{\text{OLS}}$.

2. Construct a consistent estimator $\widehat{\boldsymbol{\Sigma}}$ of $\boldsymbol{\Sigma}$ from $\widehat{\boldsymbol{\varepsilon}}$ and compute

$$\widehat{\boldsymbol{\beta}}_{\text{2WLS}} = \left(\boldsymbol{X}^\top \widehat{\boldsymbol{\Sigma}}^{-1} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \widehat{\boldsymbol{\Sigma}}^{-1} \boldsymbol{Y}.$$

Under standard regularity conditions, $\hat{\beta}_{\text{2WLS}}$ is consistent and asymptotically efficient, achieving the Cramér–Rao bound in the parametric Gaussian model and the semiparametric efficiency bound in the nonparametric setting.

Cramér–Rao bound.

- A classical Cramér–Rao lower bound exists conditional on $\boldsymbol{\Sigma}$:

$$\text{Var}(\widehat{\boldsymbol{\beta}}) \succeq (\boldsymbol{X}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{X})^{-1}.$$

- OLS does not achieve this bound unless $\boldsymbol{\Sigma} \propto I$.
- Two-step weighted least squares (2WLS), using a consistent estimator $\hat{\boldsymbol{\Sigma}}$, achieves the Cramér–Rao lower bound asymptotically.

Efficiency.

1. Use OLS residuals to estimate $\boldsymbol{\Sigma}$.
2. Incorporate $\hat{\boldsymbol{\Sigma}}^{-1}$ into weighted least squares.

0.3 Case 3: Nonparametric error distribution

Assume only the moment condition

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \mathbb{E}(\boldsymbol{\varepsilon} \mid \boldsymbol{X}) = 0,$$

with no parametric assumption on the distribution of ε.

- The model is defined by moment conditions rather than a likelihood.
- OLS is consistent under mild regularity conditions.
- The variance of $\widehat{\boldsymbol{\beta}}$ depends on an unknown covariance structure.
- Normal-based tests are asymptotically valid but not exact.

Cramér–Rao bound.

- The classical Cramér–Rao lower bound does not apply, since no likelihood is specified.
- An efficiency bound exists in the semiparametric sense.
- The semiparametric efficiency bound for β is

$$\text{Var}(\widehat{\boldsymbol{\beta}}) \succeq \left(\boldsymbol{X}^\top \boldsymbol{\Omega}^{-1} X\right)^{-1},$$

where $\boldsymbol{\Omega} = \mathbb{E}(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top \mid \boldsymbol{X})$.

Efficiency considerations.

- OLS generally does not achieve the semiparametric efficiency bound.
- Optimal weighting (e.g., GMM or 2WLS) achieves the efficiency bound when Ω is consistently estimated.
- Efficiency is defined relative to the largest class of admissible estimators under the moment restrictions.

This comparison highlights that nonparametric thinking is about relaxing assumptions while retaining valid and optimal inference through appropriate efficiency bounds.

Example 2. Conformal inference

Confidence interval for a predicted response in linear regression if error follows homoscedastic normal distribution. Assume the linear regression model

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}),$$

with i.i.d. errors. Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ be the design matrix, $\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{Y}$ the ordinary least squares estimator, and

$$s^2 = \frac{1}{n-p} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^\top \widehat{\boldsymbol{\beta}})^2$$

the unbiased estimator of σ^2.

Confidence interval for the mean response. For a new covariate vector $\boldsymbol{x}$, the mean response is $m(\boldsymbol{X}) = \mathbb{E}(Y \mid \boldsymbol{X}) = \boldsymbol{X}^\top \boldsymbol{\beta}$, with estimator $\widehat{m}(\boldsymbol{X}) = \boldsymbol{X}^\top \widehat{\boldsymbol{\beta}}$. A $100(1-\alpha)\%$ confidence interval for $m(\boldsymbol{X})$ is

$$\boxed{\boldsymbol{X}^\top \widehat{\boldsymbol{\beta}} \ \pm \ t_{1-\alpha/2,\, n-p}\, s \sqrt{\boldsymbol{x}^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{x}}}.$$

Prediction interval for a new observation. For a future observation Y_{new} at covariate $\boldsymbol{x}$, a $100(1-\alpha)\%$ prediction interval is

$$\boxed{\boldsymbol{X}^\top \widehat{\boldsymbol{\beta}} \ \pm \ t_{1-\alpha/2,\, n-p}\, s \sqrt{1 + \boldsymbol{x}^\top (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{x}}}.$$

```
# Prediction interval in linear regression (homoscedastic normal errors)
# Works for any lm() fit. Gives PI for new x, plus (optionally) CI for mean.

pred_interval_lm <- function(fit, newdata_val, level = 0.95) {
  stopifnot(inherits(fit, "lm"))

  # Create a data frame with the CORRECT column name 'x'
  # We use the name of the predictor from the model
  predictor_name <- all.vars(formula(fit))[2] # Gets "x" from y ~ x
  newdata <- data.frame(setNames(list(newdata_val), predictor_name))

  # Prediction interval
  pi <- predict(fit, newdata = newdata, interval = "prediction", level = level)
```

```
  # Confidence interval
  ci <- predict(fit, newdata = newdata, interval = "confidence", level = level)

  list(prediction_interval = as.data.frame(pi),
       mean_response_CI     = as.data.frame(ci))
}


#####Using above formula
# Prediction interval using the explicit matrix formula
# Linear regression with homoscedastic normal errors

prediction_interval_formula <- function(X, y, x_new, level = 0.95) {
  # X: design matrix (n x p), INCLUDING intercept column
  # y: response vector (n)
  # x_new: new covariate vector (p), INCLUDING intercept
  # level: confidence level

  n <- nrow(X)
  p <- ncol(X)

  x_new <- c(1, x_new)

  # OLS estimator
  beta_hat <- solve(t(X) %*% X) %*% t(X) %*% y

  # Residual variance estimate
  residuals <- y - X %*% beta_hat
  s2 <- as.numeric(t(residuals) %*% residuals) / (n - p)
  s <- sqrt(s2)

  # Predicted mean
  y_hat <- as.numeric(t(x_new) %*% beta_hat)

  # Standard error for prediction
  XtX_inv <- solve(t(X) %*% X)
  se_pred <- s * sqrt(1 + t(x_new) %*% XtX_inv %*% x_new)

  # t critical value
  alpha <- 1 - level
  tcrit <- qt(1 - alpha/2, df = n - p)

  # Prediction interval
  lower <- y_hat - tcrit * se_pred
  upper <- y_hat + tcrit * se_pred

  c(fit = y_hat, lwr = lower, upr = upper)
```

```
}

# ---------------------------
# Example
# ---------------------------
set.seed(123)

n <- 1000
x <- runif(n, -2, 2)
y <- 1 + 2*x + rnorm(n, sd = 1)

# Design matrix with intercept
X <- cbind(1, x)

# New x value (include intercept)
x_new <- c(0.5)

fit <- lm(y~x)

##Predictive Confidence interval from scratch
prediction_interval_formula(X, y, x_new, level = 0.95)

out <- pred_interval_lm(fit, newdata = x_new, level = 0.95)

out$prediction_interval   # Predictive confidence interval
out$mean_response_CI  #confidence interval for mean
```

In an asymptotic sense, this interval can be distribution-free as well, i.e., we do not need to explicitly assume normality of the errors. However, the assumption or satisfaction of normality makes it the most efficient interval.

The basic idea behind the level $(1-\alpha)100\%$ confidence interval for a new covariate value $\boldsymbol{x}$ is to construct a set $C(\boldsymbol{X})$ (as a function of the new predictor $\boldsymbol{x}$) such that

$$\mathbb{P}\big(y \in C(\boldsymbol{X}) \mid \boldsymbol{X} = \boldsymbol{x}\big) \geq 1-\alpha.$$

This is a *prediction set* for the random variable Y, not a confidence interval for an unknown parameter. Confidence intervals target parameters, whereas conformal methods target future observations.

Nonparametric (conformal) approach. In the nonparametric setting, we can fit the response using any prediction method and get $\widehat{m}(\boldsymbol{X})$ (linear regression, random forest, neural networks, etc.). Compute residuals

$$\hat{e}_i = |y_i - \widehat{m}(\boldsymbol{X})|.$$

Let $q_{1-\alpha}$ be the empirical $(1-\alpha)$ quantile of $\{\hat{e}_i\}$. The conformal prediction interval is

$$C(\boldsymbol{X}) = \big[\hat{m}(\boldsymbol{X}) - q_{1-\alpha},\ \hat{m}(\boldsymbol{X}) + q_{1-\alpha}\big].$$

Key property. The coverage

$$\mathbb{P}\big(Y_{\text{new}} \in C(X_{\text{new}})\big) \geq 1 - \alpha$$

is *exact in finite samples*, without any distributional assumptions on the errors, regardless of the regression method used. We will discuss this approach later.

Introduction to R

We will use R and R Markdown for this course (highly recommended). The examples in the lecture notes and homework assignments will be written in R. Choosing R for your homework solutions and project is highly recommended.

Introduction to R Markdown

We will use R Markdown from R Studio to

- track data analysis
- produce high-quality documents
- reproduce the results

Introduction to LaTeX

LaTeX will enable you to create PDFs directly from the R Markdown in RStudio.

Basics of R

Vectors

```
x <- c(11,218,123,36,1001)
y <- rep(1,5)
z <- seq(1,5,by=1)
y + z
```

Matrices and data frames

```
X <- cbind(x,y,z)
```

Random sampling

```
coin <- c("H","T")
set.seed(100)
samples <- sample(coin, 100, replace=TRUE)
sum(samples=="H")
```

Graphics

```
hist(z, breaks=30)
```

Using ggplot2

```
library(ggplot2)
ggplot(mtcars, aes(x=wt,y=mpg)) +
  geom_point() +
  geom_smooth() + xlab('Weight (1000 lbs)') +
    ylab("Miles/(US) gallon")
```

Repeating tasks

In addition to `for` loop, R provides `apply` and `tapply` functions to replicate code a number of times

```
apply(X,1,mean)
apply(X,2,mean)
```

User-defined functions

```
mSummary <- function(X) {
  q1 <- quantile(x,.25)
  q3 <- quantile(x,.75)
  list(med=median(X),iqr=q3-q1)
}
```

Monte Carlo simulation

```
X <- matrix(rnorm(10*100),ncol=10)
xbar <- apply(X,1,mean)
var(xbar)
```

compared to theoretical results: $\frac{\sigma^2}{n} = \frac{1}{10}$.