Linear Models

Group 4: Abhinav, Sanket, Utkarsha, Kristin
Data Sciences in the Life Sciences

Table of contents

Introduction to Linear Models

$$Y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_2 x_{i,p} + \varepsilon_i, i = 1, \ldots, n$$

$$Y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} + \varepsilon_i, i = 1, \ldots, n$$

- For a model to be linear all of its parameters must be linear.
- The dependent variable and independent variables may be non-linear (for example, a parabolic model is considered linear despite having exponential independent variable)

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \varepsilon, i = 1, \ldots, n$$

Estimating Parameters (the Betas)

- For any given data, virtually infinite amounts of models can be prepared.
- However, for the model to be useful all the parameters (betas) must be estimated.
- Furthermore, the model must be tested whether it is a good fit or not. For this the distance between the data points and the model must be minimised.
- This distance can be calculated using the least squares equation:

$$\hat{\beta}$$

$$\sum_{i=1}^{n} \left\{ Y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \right) \right\}^2$$

Estimating parameters continued..

$$\sum_{i=1}^{n} \left\{ Y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \right) \right\}^2$$

- The minimum distance parameters calculated from the above equations are known as least square estimates and denoted using $\hat{\beta}$
- The residual sum of squares (RSS) is a square of sum of all these distances.
- lm() function in R can be used to obtain a model fit for the data. It calculates the estimate values for us along with several different other parameters.

Linear Models using R (understanding the maths behind)

The linear model can be written in matrix form as:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon, i = 1, \ldots, N \quad \Rightarrow \quad \mathbf{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \text{ and } \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}$$

Simplifying further we get: $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$

The least squares equation can also be simplified further:

$$\sum_{i=1}^{n} \left\{ Y_i - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \right) \right\}^2 \quad \Rightarrow \quad (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) \quad \text{... equation (1)}$$

- We can use calculus to find the minimum.
- Taking derivative of equation 1 and simplifying further we get:

$$2\mathbf{X}^{\mathsf{T}}(\mathbf{Y}-\mathbf{X}\hat{\boldsymbol{\beta}})=0 \quad \Rightarrow \quad \mathbf{X}^{\mathsf{T}}\mathbf{X}\hat{\boldsymbol{\beta}}=\mathbf{X}^{\mathsf{T}}\mathbf{Y} \quad \Rightarrow \quad \hat{\boldsymbol{\beta}}=(\mathbf{X}^{\mathsf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{Y}$$

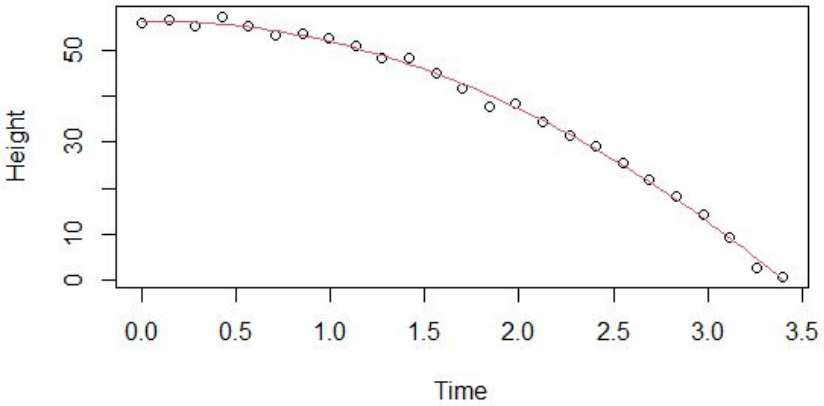
- The equation thus obtained can be used in many data analysis situations
- For example:

```
set.seed(1)
g <- 9.8 #meters per second
n <- 25
tt <- seq(0,3.4,len=n) #time in secs, t is a base function
d <- 56.67  - 0.5*g*tt^2 + rnorm(n,sd=1)
```

We can use this code to simulate measurements of a free falling object with random errors

- Then we can calculate the least square estimates for the data and plot the model using R

```
X <- cbind(1,tt,tt^2)
y <- d
betahat <- solve(crossprod(X))%*%crossprod(X,y)
newtt <- seq(min(tt),max(tt),len=100)
X <- cbind(1,newtt,newtt^2)
fitted <- X%*%betahat
plot(tt,y,xlab="Time",ylab="Height")
lines(newtt,fitted,col=2)
```



- The calculated least square estimates are saved as betahat:

```
##              [,1]
##     56.5317368
## tt   0.5013565
##     -5.0386455
```

intercept

time taken

$(\text{time taken})^2$

$$\mathbf{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \text{ and } \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}$$

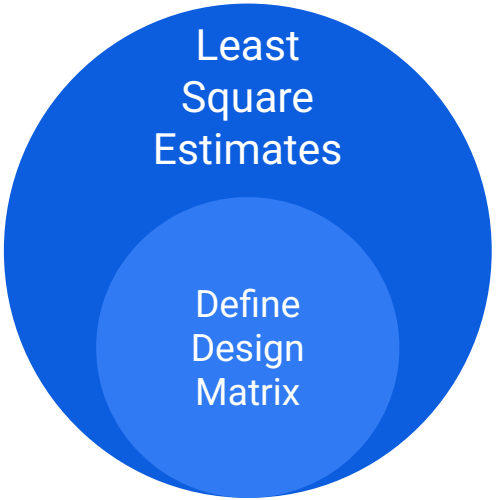
$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}$$

Why choice of design is important ?

Encoding which coefficients will fit

Inter-relationship between the samples

Fitting the model



Myths ?

- Follows straightforward dictum of samples
- Basic information ≠ “Correct” design

Which design works ?

For the comparison between different groups ≧ 2 columns:

1. Intercept Column (contains 1's)
2. Second Column (specifies which samples are second group)

Population average of the 1st group

Difference in population of the 1st and the 2nd group

Our Interest !
Why ?

Encoding in R

factor() is an instrument to indicate the membership of the group by assigned 0's and 1's.

Note: names of the levels are irrelevant, but the order matters!

```
diet <- factor(c(1,1,1,1,2,2,2,2))
sex <- factor(c("f","f","m","m","f","f","m","m"))
table(diet,sex)
```

```
group <- factor(c(1,1,2,2,3,3))
model.matrix(~ group)
```

Need more groups ?

- Ensure separate coefficients for each group

Need more variables ?

- factor -> group combinations
- Use of operators (+ , : , *) in *model.matrix()*.

Releveling

The level which is contrasted against is called **Reference level.**

If we want any other group to be the reference level except the default one:

Use *relevel()* function

or

Provide explicit levels to the *factor()* call

model.matrix() grabs the variable from the R Global Environment unless our data is passed explicitly as a *data.frame()* in the *data()* argument.

```
group <- factor(c(1,1,2,2))
group <- relevel(group, "2")
model.matrix(~ group)
```

```
group <- factor(group, levels=c("1","2"))
model.matrix(~ group)
```

Continuous Vs Indicator Variables

```
tt <- seq(0,3.4,len=4)
model.matrix(~ tt + I(tt^2))
```

```
##   (Intercept)       tt   I(tt^2)
## 1           1 0.000000  0.000000
## 2           1 1.133333  1.284444
## 3           1 2.266667  5.137778
## 4           1 3.400000 11.560000
## attr(,"assign")
## [1] 0 1 2
```

used as **arithmetically**, in *model.matrix()*.

Indicator variables assume a different mean between two groups.

Continuous variables assume a very specific relationship between outcome and predictor values.

Why are we interested in using the continuous over indicator ?

Example : Testing the dosages of treatment to find a specific relationship between a measured quantity and the dosage.

The Mathematics behind lm() function

How well your model fits the data ?

Normalized using sample size and # of variables

“Global” test to check at least one of the coefficients ≠ 0

```
lm <- function (formula, data, subset, weights, na.action,
                method = "qr", model = TRUE, x = FALSE, y = FALSE,
                qr = TRUE, singular.ok = TRUE, contrasts = NULL,
                offset, ...)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-6.1042 -2.4358 -0.4138  2.8335  7.1858

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   23.813      1.039  22.912   <2e-16 ***
Diethf         3.021      1.470   2.055   0.0519 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.6 on 22 degrees of freedom
Multiple R-squared:  0.1611,        Adjusted R-squared:  0.1229
F-statistic: 4.224 on 1 and 22 DF,  p-value: 0.05192
```

```r
Y <- dat$Bodyweight
X <- model.matrix(~ Diet, data=dat)
solve(t(X) %*% X) %*% t(X) %*% Y
```

The t-test formula for the standard error of the difference, if we assume equal variance in the two groups, is the square root of the variance:

$$\frac{1}{1/N_x + 1/N_y} \frac{\sum_{i=1}^{N_x} (X_i - \mu_x)^2 + \sum_{i=1} (Y_i - \mu_y)^2}{N_x + N_y - 2}$$

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

FORTRAN

F77_CALL in lm.c calls the function dqlrs in which the LINPACK routines to compute the QR decomposition using household reflections of an n by p matrix x. compute coordinate transformations, projections, and least squares solutions.
Convoluted or spaghetti code!

C

Regression gets calculated and .Call calls into the C code (lm.c). The function Cdqrls:

concerned with checking invariants of inputs, and constructing and initializing new objects.

R

Constructing the design matrix:

lm() conveniently works with formulas and data.frames, lm.fit() wants matrices, so moving from lm() to lm.fit() removes one layer of abstraction.

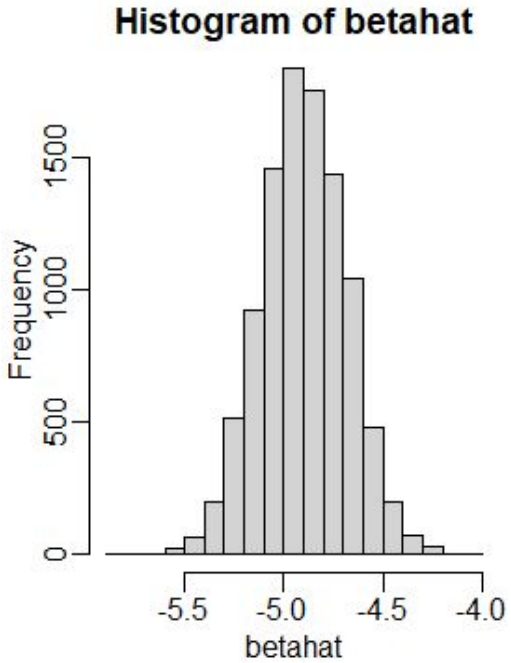
Standard Errors

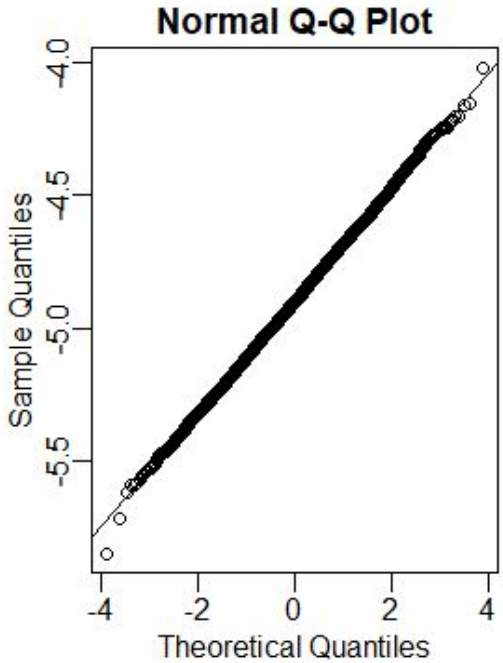
Introduction:

- It is useful to think about where variance comes from.
- It can arise due to measuring errors, inherent randomness of a system, external factors, etc.
- For example, in the falling object example it comes from measurement errors.
- Hence, every time the experiment is performed we will get different estimates.

Delving deeper in the falling object model

- Running a Monte Carlo simulation on this model will give a different estimate every time





- These plots represent the distribution of the calculated estimates after performing a Monte Carlo simulation.
- Furthermore, mean of this distribution is equal to the true parameter (-0.5g or -4.9)

Variance-covariance matrix

- Assume that you have a vector of random variables Y. Then its covariance matrix Σ with i, j observations will be $\Sigma_{i,j} \equiv \mathrm{Cov}(Y_i, Y_j)$
- Here,
  - If i = j, covariance = variance, $\mathrm{Cov}(Y_i, Y_i) = \mathrm{var}(Y_i) = \sigma^2$, therefore $\sum = \sigma^2 I$, where I is the identity matrix.
  - If i and j are independent, covariance = 0, $\mathrm{Cov}(Y_i, Y_j) = 0, \text{ for } i \neq j$

Variance of a linear combination

- Variance-covariance matrix of a linear combination of AY of Y can be calculated using: $\mathrm{var}(\mathbf{AY}) = \mathbf{A}\mathrm{var}(\mathbf{Y})\mathbf{A}^\top$
- Furthermore, if Y_1 and Y_2 are independent then:

$$\mathrm{var}\{Y_1 + Y_2\} = \mathrm{var}\left\{\begin{pmatrix}1 & 1\end{pmatrix}\begin{pmatrix}Y_1\\Y_2\end{pmatrix}\right\} \Rightarrow = \begin{pmatrix}1 & 1\end{pmatrix}\sigma^2\mathbf{I}\begin{pmatrix}1\\1\end{pmatrix} = 2\sigma^2$$

- $\hat{\beta}$ is linear combination of Y : AY with $A = (X^TX)^{-1}X^T$ and,

$$\mathrm{var}(\hat{\boldsymbol{\beta}}) = \mathrm{var}((\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}) = \sigma^2(\mathbf{X}^\top\mathbf{X})^{-1}$$

Missing piece of the puzzle: σ^2

- We need residuals to calculate σ^2, and residuals = $\mathbf{r} \equiv \hat{\boldsymbol{\epsilon}} = \mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}$

- Which can then be used in the formula: $s^2 \equiv \hat{\sigma}^2 = \frac{1}{N-p}\mathbf{r}^\top\mathbf{r} = \frac{1}{N-p}\sum_{i=1}^{N} r_i^2$

Analysis of Covariance (ANCOVA)

Input Data

```
input <- mtcars[,c("am","mpg","hp")]
print(head(input))
```

When we execute the above code, it produces the following result –

```
                  am   mpg   hp
Mazda RX4         1    21.0  110
Mazda RX4 Wag     1    21.0  110
Datsun 710        1    22.8   93
Hornet 4 Drive    0    21.4  110
Hornet Sportabout 0    18.7  175
Valiant           0    18.1  105
```

ANOVA Analysis

We create a regression model taking "hp" as the predictor variable and "mpg" as the response variable taking into account the interaction between "am" and "hp".

Model with interaction between categorical variable and predictor variable

```
# Get the dataset.
input <- mtcars

# Create the regression model.
result <- aov(mpg~hp*am,data = input
print(summary(result))
```

When we execute the above code, it produces the following result –

```
            Df Sum Sq Mean Sq F value   Pr(>F)
hp           1  678.4   678.4  77.391 1.50e-09 ***
am           1  202.2   202.2  23.072 4.75e-05 ***
hp:am        1    0.0     0.0   0.001    0.981
Residuals   28  245.4     8.8
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

This result shows that both horse power and transmission type has significant effect on miles per gallon as the p value in both cases is less than 0.05. But the interaction between these two variables is not significant as the p-value is more than 0.05.

Model without interaction between categorical variable and predictor variable

```
# Get the dataset.
input <- mtcars

# Create the regression model.
result <- aov(mpg~hp+am,data = input)
print(summary(result))
```

When we execute the above code, it produces the following result −

```
            Df  Sum Sq  Mean Sq   F value   Pr(>F)
hp           1  678.4   678.4   80.15 7.63e-10 ***
am           1  202.2   202.2   23.89 3.46e-05 ***
Residuals   29  245.4     8.5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

This result shows that both horse power and transmission type has significant effect on miles per gallon as the p value in both cases is less than 0.05

Comparing Two Models

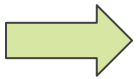
```
# Get the dataset.
input <- mtcars

# Create the regression models.
result1 <- aov(mpg~hp*am,data = input)
result2 <- aov(mpg~hp+am,data = input)

# Compare the two models.
print(anova(result1,result2))
```

When we execute the above code, it produces the following result –

```
Model 1: mpg ~ hp * am
Model 2: mpg ~ hp + am
  Res.Df    RSS Df  Sum of Sq     F Pr(>F)
1     28 245.43
2     29 245.44 -1 -0.0052515 6e-04 0.9806
```

Co-linearity

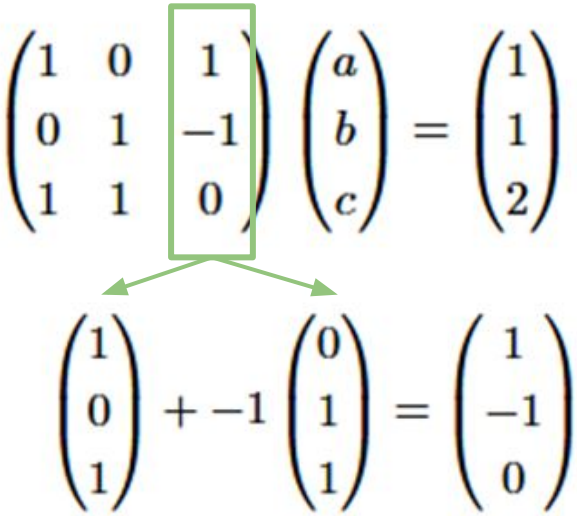
change in one independent variable X_j by 1 unit **when holding all other independent variables constant!**



mean change in dependent variable by β_j units

Co-linearity occurs when some of the independent variables are not independent from each other i.e. they are correlated.

Matrix Algebra behind Co-linearity



The third column is collinear with the first two columns i.e. it can be written as a linear combination of the other columns.

Matrix Algebra behind Co-linearity

$$\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + b \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} + c \begin{pmatrix} 1-0 \\ 0-1 \\ 1-1 \end{pmatrix}$$

$$= (a+c) \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + (b-c) \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

> infinite number of solutions

Matrix Algebra behind Co-linearity

- Design Matrix X

$$\mathbf{X} = \begin{pmatrix} \mathbf{1} & \mathbf{X}_1 & \mathbf{X}_2 & \mathbf{X}_3 \end{pmatrix} \text{ with } \mathbf{X}_3 = -\mathbf{X}_2$$

- Residuals

$$\mathbf{Y} - \{\mathbf{1}\beta_0 + \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2 + \mathbf{X}_3\beta_3\} = \mathbf{Y} - \{\mathbf{1}\beta_0 + \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2 - \mathbf{X}_2\beta_3\}$$
$$= \mathbf{Y} - \{\mathbf{1}\beta_0 + \mathbf{X}_1\beta_1 + \mathbf{X}_2(\beta_2 - \beta_3)\}$$

if $\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$ is a least squares solution, then, for example, $\hat{\beta}_1, \hat{\beta}_2 + 1, \hat{\beta}_3 + 1$ is also a solution.

Effects of Co-linearity

- coefficients become very sensitive to small changes in the model and hard to interprete
- reduces precision of estimated coefficients and statistical power
- but: does not affect the prediction of the model, only the coefficient interpretability

$$\begin{pmatrix} Sex & A & B & C & D \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

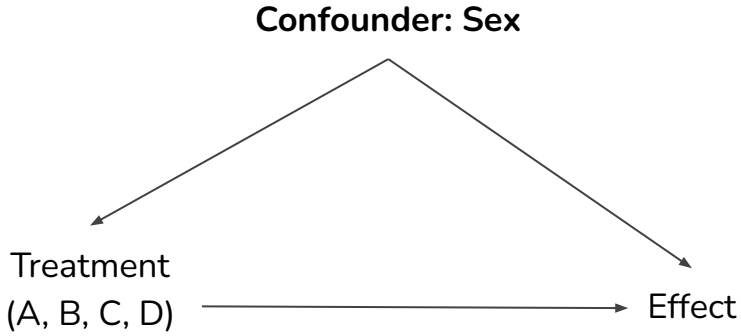
ıfounding variables

ıfluence both the dependent variable and independen

xample study design: 4 treatments (A, B, C, D) and tw

er study group

$$\begin{pmatrix} Sex \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} C \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} D \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$



Removing Confounders

$$\begin{pmatrix} Sex & A & B & C & D \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

- small changes in study design can remove the Co-Linearity within the matrix

  > unique least square solution exists

Removing Confounders

- rank of a matrix A is defined as the number of linear independent columns of A
- if Rank(A) < NumberOfColumns(A), then the LSE are not unique
- in R: qr()$rank function returns the rank of a matrix

$$A = \begin{pmatrix} Sex & A & B & C & D \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

rank(A) = 4

$$B = \begin{pmatrix} Sex & A & B & C & D \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

rank(B) = 5

Matrix A has no unique LSE, matrix B has!

Thank you for paying attention!

Any questions?