Data Visualization

**Lecture5**

**MathamaticsVisualization**

Phuc Loi Luu, PhD

p.luu@garvan.org.au

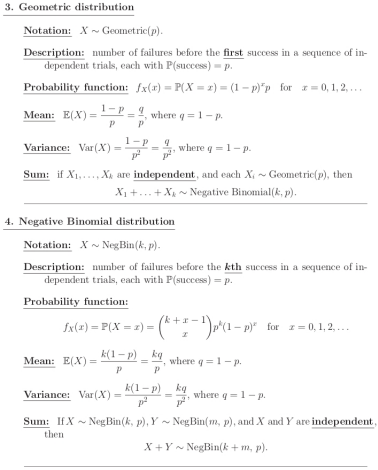
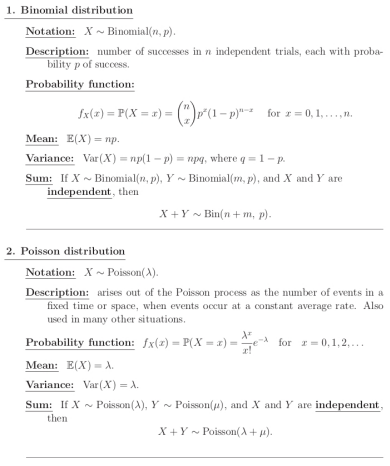
luu.p.loi@googlemail.com

1

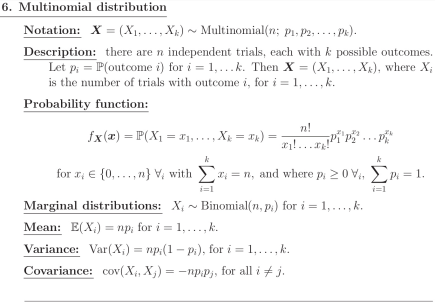
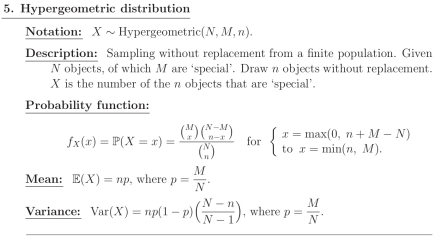
Content

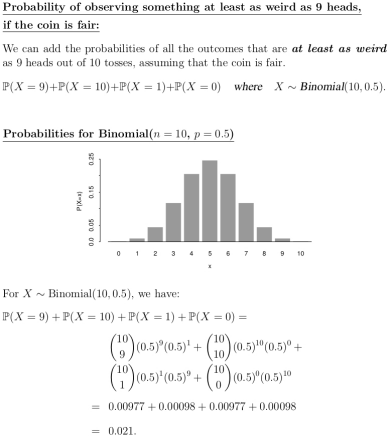
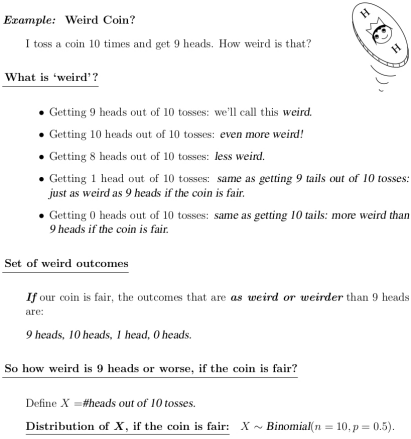
Plot a mathmatics graph • Introduction to Probability and Stastitics• Create a ShinyApp

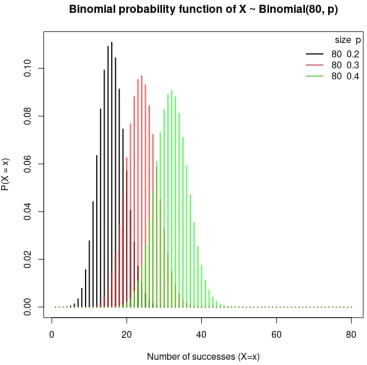
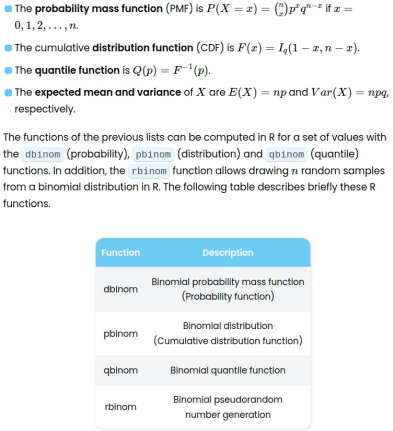
2

**Some important discrete distribution **3

**Some important discrete distribution**

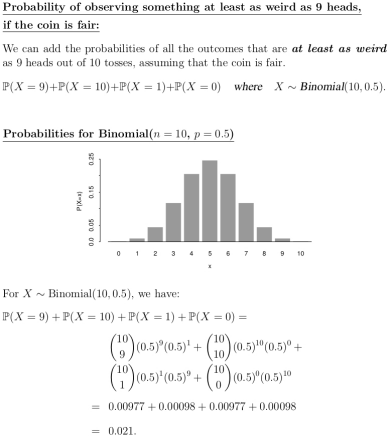
****4

**Revisit: Hypothesis testing**5

**The binomial distribution **

6

**The binomial distribution**

The R command, **dbinom**functioncalculatethe

binomial probability function for a set of valuesX=x,a

number of trials n and a probability of successp:

dbinom(x=0, size=10, prob=0.5)=P(X=0)=0.000977

where X ∼ Binomial(10, 0.5)

dbinom(x=0, size=10, prob=0.5)

# [1] 0.0009765625

dbinom(x=1, size=10, prob=0.5)

# [1] 0.009765625

dbinom(x=9, size=10, prob=0.5)

# [1] 0.009765625

dbinom(x=10, size=10, prob=0.5)

# [1] 0.0009765625

x <- c(0, 1, 9, 10)

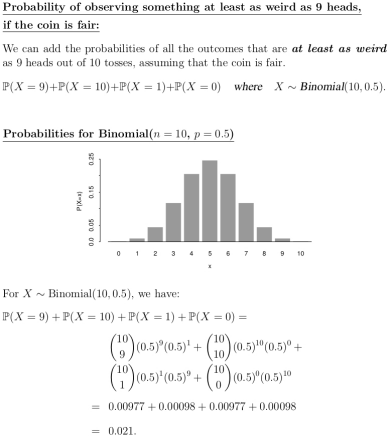
sum(sapply(x, function(x) dbinom(x=x, size=10,

prob=0.5)))

# [1] 0.02148438

7

**The binomial probabilityfunction**

The R command, **dbinom** function calculatethebinomial

probability function for a set of values X=x, a numberoftrialsn

and a probability of success p:

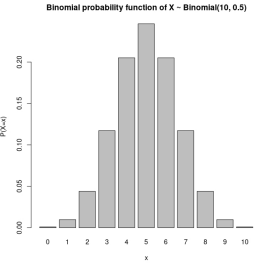
*x <- seq(0, 10, 1)*

*d <- dbinom(x=x, size=10, prob=0.5)*

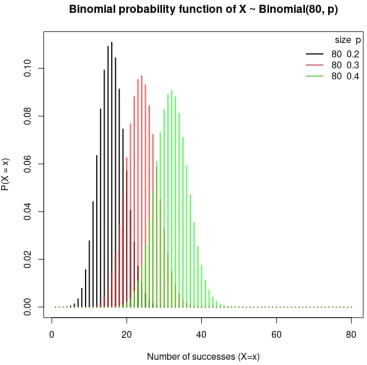
*names(d) <- x*

*barplot(d, main="Binomial probability functionofX~*

*Binomial(10, 0.5)", ylab="P(X=x)", xlab="x")*

**8

**The binomial probabilityfunction**

# The R command, plot Binomial probability function 

*# Grid of X-axis values*

*x <- 1:80*

*# size=80, prob=0.2*

*plot(dbinom(x, size=80, prob=0.2), type="h", lwd=2,*

*main="Binomial probability function of X ~ Binomial(80, p)",*

*ylab="P(X=x)", xlab="Number of successes (X=x)")*

*# size=80, prob=0.3*

*lines(dbinom(x, size=80, prob=0.3), type="h",*

*lwd=2, col=rgb(1,0,0, 0.7))*

*# size=80, prob=0.4*

*lines(dbinom(x, size=80, prob=0.4), type="h",*

*lwd=2, col=rgb(0, 1, 0, 0.7))*

*# Add a legend*

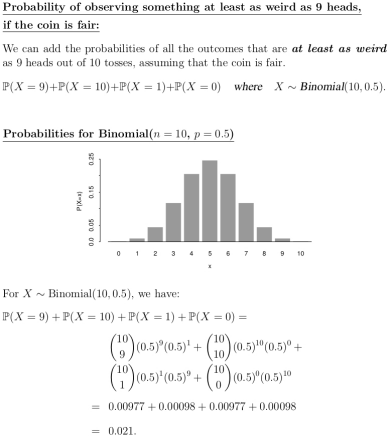
*legend("topright", legend=c("80 0.2", "80 0.3", "80 0.4"),*

*title="size p", title.adj=0.95,*

*lty=1, col=1:3, lwd=2, box.lty=0)*

https://r-coder.com/binomial-distribut9ion-r/

**Hypothesis testing withR**

• The R command, **pbinom**calculate theprobabilityof

a variable X following a binomial distributiontaking

values lower than or equal to x

• The **pbinom** is equivalent to the **cumulative**

**di s t r ibut i on func t i on** f o r t heBi nomial

distribution:

▪ pbinom(q=0, size=10, prob=0.5)=P(X≤0)=FX(0)

where X ∼ Binomial(10, 0.5)

*pbinom(q=0, size=10, prob=0.5)*

*# [1] 0.0009765625*

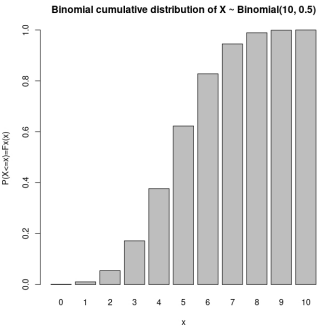
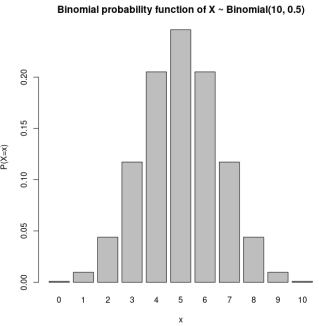
▪ For X ∼ Binomial(10, 0.5). The overall p-valuein

this example is 2 \* FX (1).

*p\_value <- 2\*pbinom(q=1, size=10, 0.5)*

*# [1] 0.02148438*

10

**The binomial cumulative function**

• Binomial cumulative distribution of X ~ Binomial(10, 0.5)

*x <- seq(0, 10, 1)*

*acc <- pbinom(q=x, size=10, prob=0.5)*

*names(acc) <- x*

*barplot(acc, main="Binomial cumulative distribution of X ~ Binomial(10, 0.5)", ylab="P(X<=x)=Fx(x)", xlab="x")* 11

**The binomial cumulativefunction**

• Binomial cumulative distribution of X ~ Binomial(80, p) 

*# Grid of X-axis values*

*x <- 1:80*

*# size=80, prob=0.2*

*plot(pbinom(x, size=80, prob=0.2), type="s", lwd=2,*

*main="Binomial distribution function of X ~ Binomial(80, p)",*

*xlab="Number of successes (X=x)", ylab="P(X=x)=Fx(x)")*

*# size=80, prob=0.3*

*lines(pbinom(x, size=80, prob=0.3), type="s", lwd=2, col=2)*

*# size=80, prob=0.4*

*lines(pbinom(x, size=80, prob=0.4), type="s", lwd=2, col=3)*

*# Add a legend*

*legend("bottomright", legend=c("80 0.2", "80 0.3", "80 0.4"),*

*title="size p", title.adj=0.95, lty=1, col=1:3, lwd=2, box.lty=0)*

12

**The binomial cumulativefunction**# size: number of trials (n >=0)

# prob: probability of success on each trial # lb: lower bound of the sum # ub: upper bound of the sum # col: color

# lwd: line width

P(X<=6)=Fx(X=6)=F(6) *pbinom(q=6, size=20, prob=0.4)*

binom\_sum <- function(size, prob, lb, ub, col=4, lwd=1, ...) {

x <- 0:size

if (missing(lb)) { lb <- min(x)

}

if (missing(ub)) { ub <- max(x)

}

*binom\_sum(size=20, prob=0.4, ub=6, lwd=2,ylab="P(X=x)", xlab="Numberof successes")*

plot(dbinom(x, size=size, prob=prob), type="h", lwd=lwd, ...) 

if(lb == min(x) & ub == max(x)) {

color <- col

} else {

color <- rep(1, length(x))

color[(lb + 1):ub ] <- col

}

lines(dbinom(x, size=size, prob=prob), type="h",

col= color, lwd=lwd, ...)

}

13

**The binomial cumulativefunction**

# size: number of trials (n >=0)

# prob: probability of success on each trial # lb: lower bound of the sum # ub: upper bound of the sum # col: color

# lwd: line width

P(X<6)=P(X<=5)=Fx(X=5)=F(5)

*pbinom(q=5, size=20, prob=0.4) 1 - pbinom(q=5, size=20, prob=0.4, lower.tail=FALSE)sum(dbinom(0:5, size=20, prob=0.4))*

binom\_sum <- function(size, prob, lb, ub, col=4, lwd=1, ...) {

x <- 0:size

if (missing(lb)) { lb <- min(x)

}

if (missing(ub)) { ub <- max(x)

}

*binom\_sum(size=20, prob=0.2, lwd=2, col=2, ub=5,ylab="P(X=x)", xlab="Number of successes")*

plot(dbinom(x, size=size, prob=prob), type="h", lwd=lwd, ...) 

if(lb == min(x) & ub == max(x)) {

color <- col

} else {

color <- rep(1, length(x))

color[(lb + 1):ub ] <- col

}

lines(dbinom(x, size=size, prob=prob), type="h",

col= color, lwd=lwd, ...)

}

14

**The binomial cumulative function**

# size: number of trials (n >=0)

# prob: probability of success on each trial # lb: lower bound of the sum # ub: upper bound of the sum # col: color

# lwd: line width

P(7≤X≤11)=?

*pbinom(q=11, size=20, prob=0.4) - pbinom(q=7, size=20,prob=0.4)*

*sum(dbinom(7:11, size=20, prob=0.4))*

binom\_sum <- function(size, prob, lb, ub, col=4, lwd=1, ...) {

x <- 0:size

if (missing(lb)) { lb <- min(x)

}

if (missing(ub)) { ub <- max(x)

}

*binom\_sum(size=20, prob=0.4, lb=7, ub=11, lwd=2, ylab="P(X=x)", xlab="Number of successes")*

plot(dbinom(x, size=size, prob=prob), type="h", lwd=lwd, ...) 

if(lb == min(x) & ub == max(x)) {

color <- col

} else {

color <- rep(1, length(x))

color[(lb + 1):ub ] <- col

}

lines(dbinom(x, size=size, prob=prob), type="h",

col= color, lwd=lwd, ...)

}

15

**The binomial cumulative function, exercise1**

# size: number of trials (n >=0)

# prob: probability of success on each trial # lb: lower bound of the sum # ub: upper bound of the sum # col: color

# lwd: line width

For X ∼ Binomial(10, 0.5). The overall p-valueinthis example is 2 \* FX (1).

*2\*pbinom(q=1, size=10, prob=0.5)*

binom\_sum <- function(size, prob, lb, ub, col=4, lwd=1, ...) {

x <- 0:size

if (missing(lb)) { lb <- min(x)

}

if (missing(ub)) { ub <- max(x)

}

*# [1] 0.02148438*

*binom\_sum(size=10, prob=0.5, lwd=2, col=2,lb=?,ub=?, ylab="P(X=x)", xlab="Number of successes")*

**Exercise 1:** rewrite binom\_sumfunctiontovisualize P(X<=1) and P(X>=9) inthesameplot

plot(dbinom(x, size=size, prob=prob), type="h", lwd=lwd, ...)

if(lb == min(x) & ub == max(x)) {

color <- col

} else {

color <- rep(1, length(x))

color[(lb + 1):ub ] <- col

}

lines(dbinom(x, size=size, prob=prob), type="h",

col= color, lwd=lwd, ...)

}

16

**Binomial Distribution: generaterandom**

The rbinom function allows you to draw nn random observations from a binomial distribution.

Example, generate 15 random observations from a binomial distribution if the

number of trials is 30 and the probability of success on each trial is 0.1

set.seed(2)

rbinom(n=15, size=30, prob=0.1)

[1] 2 4 3 1 6 6 1 5 3 3 3 2 4 1 2

library(ggplot2)

library(reshape2)

# Data

set.seed(5)

x <- cbind(N=””, =rnorm(10), n100=rnorm(100), n10=rnorm(1000))

df <- data.frame(x)

# Histogram with kernel density

ggplot(df, aes(x = x)) +

geom\_histogram(aes(y = ..density..),

colour = 1, fill = "white") +

geom\_density(lwd = 1, colour = 4,

fill = 4, alpha = 0.25)

17

**Examples of Conditional Expectation and Variance**

****18

**Examples of Conditional Expectation and Variance **19

**Examples of Conditional Expectation and Variance **20

**Examples of Conditional Expectation and Variance **

21

**Continuous Random Variables **22

**Normal Distribution** *X ~ N(mean, variance)*

**23

**Normal Distribution: probability function**

par(mfrow=c(1, 2))

# Grid of X-axis values

x <- seq(-4, 8, 0.1)

#-----------------------------------------

# Same standard deviation, different mean

#-----------------------------------------

# Mean 0, sd 1

plot(x, dnorm(x, mean=0, sd=1), type="l", 

ylim=c(0, 0.6), ylab="", lwd=2, col="red")

# Mean 3, sd 1

lines(x, dnorm(x, mean=3, sd=1), col="blue", lty=1, lwd=2)

# Adding a legend

legend("topright", legend=c("0 1", "3 1"), col=c("red", "blue"),

title=expression(paste(mu, " ", sigma)),

title.adj=0.9, lty=1, lwd=2, box.lty=0)

#-----------------------------------------

# Same mean, different standard deviation

#-----------------------------------------

# Mean 1, sd 1

plot(x, dnorm(x, mean=1, sd=1), type="l",

ylim=c(0, 1), ylab="", lwd=2, col="red")

# Mean 1, sd 0.5

lines(x, dnorm(x, mean=1, sd=0.5), col="blue", lty=1, lwd=2)

# Adding a legend

legend("topright", legend=c("1 1", "1 0.5"), col=c("red", "blue"),

title=expression(paste(mu, " ", sigma)),

title.adj=0.75, lty=1, lwd=2, box.lty=0)

par(mfrow=c(1, 1))

https://r-coder.com/normal-distribution-r/24

**Normal Distribution**

# Normal probability distribution plot

Mean <- 1000 

Sd <- 10

# X grid for non-standard normal distribution

x <- seq(-3, 3, length=100) \* Sd + Mean

# Density function

f <- dnorm(x, Mean, Sd)

plot(x, f, type="l", lwd=2, col="blue", ylab="", xlab="Weight")

abline(v=Mean) # Vertical line on the mean

lb <- min(x) # Lower bound

ub <- 1010 # Upper bound

x2 <- seq(min(x), ub, length=100) # New Grid

y <- dnorm(x2, Mean, Sd) # Density

plot(x, f, type="l", lwd=2, col="blue", ylab="", xlab="Weight")

abline(v=ub)

polygon(c(lb, x2, ub), c(0, y, 0), col=rgb(0, 0, 1, alpha=0.5))

text(995, 0.01, "84.13%")

25

**Normal Distribution**

# mean: mean of the Normal variable

# sd: standard deviation of the Normal variable

# lb: lower bound of the area

# ub: upper bound of the area

# acolor: color of the area

# ...: additional arguments to be passed to lines function

normal\_area(mean=0, sd=1, lb=-1, ub=2, lwd=2)

normal\_area <- function(mean=0, sd=1, lb, ub, acolor="lightgray", ...) { 

x <- seq(mean - 3 \* sd, mean + 3 \* sd, length=100)

if (missing(lb)) {

lb <- min(x)

}

if (missing(ub)) {

ub <- max(x)

}

x2 <- seq(lb, ub, length=100)

plot(x, dnorm(x, mean, sd), type="n", ylab="")

y <- dnorm(x2, mean, sd)

polygon(c(lb, x2, ub), c(0, y, 0), col=acolor)

lines(x, dnorm(x, mean, sd), type="l", ...)

}

26

**Normal Distribution**

\* Probability of a box weighing more than 980 grams 

P(X > 980)=P(X >= 980)

*pnorm(980, Mean, Sd, lower.tail=FALSE) # 0.9772499 or 97.72%*

*1 - pnorm(980, Mean, Sd) # Equivalent*

*pnorm(1020, Mean, Sd) # Equivalent by symmetry*

*normal\_area(mean=Mean, sd=Sd, lb=980, acolor=rgb(0, 0, 1, alpha=0.5))*

*text(1000, 0.01, "97.72%")*

**

\* Probability of a box weighing more than 990 grams and less than 1000 you have

to calculate ?

P(990 < X < 1000)

pnorm(1000, Mean, Sd) - pnorm(990, Mean, Sd) # 0.3413447 or 34.13%

normal\_area(mean=Mean, sd=Sd, lb=990, ub=1000, acolor=rgb(0, 0, 1, alpha=0.5))

text(995, 0.01, "34.13%", srt=90)

27

*par(mfrow=c(1, 2))*

**Normal Distribution: Plot cumulative distribution***#-----------------------------------------*

*# Grid of X-axis values*

*x <- seq(-4, 8, 0.1)*

*#-----------------------------------------*

*# Same standard deviation, different mean*

*#-----------------------------------------*

*# Mean 0, sd 1*

*plot(x, pnorm(x, mean=0, sd=1), type="l", ylim=c(0, 1), ylab="", lwd=2, col="red") # Mean 3, sd 1*

*lines(x, pnorm(x, mean=3, sd=1), col="blue", lty=1, lwd=2)*

*# Legend*

*legend("topleft", legend=c("0 1", "3 1"), col=c("red", "blue"),*

*title=expression(paste(mu, " ", sigma)),*

*title.adj=0.9, lty=1, lwd=2, box.lty=0)*

**

*# Same mean, different standard deviation #-----------------------------------------*

*# Mean 1, sd 1*

*plot(x, pnorm(x, mean=1, sd=1), type="l",*

*ylim=c(0, 1), ylab="", lwd=2, col="red")*

*# Mean 1, sd 0.5*

*lines(x, pnorm(x, mean=1, sd=0.5), col="blue", lty=1, lwd=2)*

*# Legend*

*legend("topleft", legend=c("1 1", "1 0.5"), col=c("red", "blue"), title=expression(paste(mu, " ", sigma)),*

*title.adj=0.75, lty=1, lwd=2, box.lty=0)*

*par(mfrow=c(1, 1))*

28

**Normal Distribution: Plot cumulative distribution**

P(X<0)=0.5 for a standard Normal distribution 

x <- seq(-4, 4, 0.1)

plot(x, pnorm(x, mean=0, sd=1), type="l", ylim=c(0, 1), ylab="P(X < x)", lwd=2, col="red")

segments(0, 0, 0, 0.5, lwd=2, lty=2)

segments(-4, 0.5, 0, 0.5, lwd=2, lty=2)

29

**Normal Distribution: quantile for probability**

• The qnorm function allows you to find the quantile (percentile)

Q for any probability p.

• The qnorm function is the inverse of the pnorm function 

• The quantile for probability 0.5 (Q(0.5)) on a symmetric

distribution is equal to the mean

*qnorm(0.5, mean=0, sd=1) # 0*

*x <- pnorm(q=-1.5, mean=0, sd=1) # 0.0668072*

*qnorm(x, mean=0, sd=1) # -1.5*

*normal\_area(mean=0, sd=1, ub=-1.5, lwd=2, acolor=rgb(0, 0,*

*1, alpha=0.5))*

*arrows(-0.5, 0.1, -1.45, 0, lwd=2, length=0.2)*

*text(-0.25, 0.13, "-1.5", cex=1.5)*

30

**Normal Distribution: generate random**

par(mfrow=c(1, 3))

• The **rnorm** function generates nn observations from the Normal distribution with mean μ and standard deviation σ

x <- seq(-10, 10, length=200)

• Generate 10 observations of a standard Normal distribution set.seed(3)

*set.seed(1) rnorm(10)*

# n=10

hist(rnorm(10, mean=0, sd=1), main="n=10", xlab="", prob=TRUE)

lines(x, dnorm(x), col="red", lwd=2)

*[1] -0.19250276 1.25912983 -0.53794513 -0.76502724*

*0.23342112 -1.47363447*

# n=100

hist(rnorm(100, mean=0, sd=1), main="n=100",

*[7] 0.04490047 0.99764774 -1.95028133 -0.68274247*

xlab="", prob=TRUE)

lines(x, dnorm(x), col="red", lwd=2)

# n=1000

hist(rnorm(1000, mean=0, sd=1), main="n=1000",

xlab="", prob=TRUE)

lines(x, dnorm(x), col="red", lwd=2) 

par(mfrow=c(1, 1))

31

**Normal Distribution: One Sample Z-Test inR**

• Suppose the IQ in a certain population is normally distributed

# install.packages("BSDA")

# Z test for mean of one sample

with a mean of μ = 100 and standard deviation of σ = 15.

library(BSDA)

mu <- 100

• A scientist wants to know if a new medication affects IQ levels, so

sigma <- 15

# enter IQ levels for 20 patients

she recruits 20 patients to use it for one month and records their data <- c(88, 92, 94, 94, 96, 97, 97, 97, 99, 99,

IQ levels at the end of the month. 

105, 109, 109, 109, 110, 112, 112, 113, 114, 115)

# perform one sample z-test using z.test in BSDA packagez.test(data, mu=mu, sigma.x=sigma)

# One-sample z-Test

# data: data

# z = 0.90933, p-value = 0.3632 # alternative hypothesis: true mean is not equal to 100# 95 percent confidence interval:

# 96.47608 109.62392

# sample estimates:

# mean of x

# 103.05

# perform one sample z-test manually Zst <- (mean(data)-mu)/(sigma/sqrt(length(data))) dtp <- 1 - 2\*(1-pnorm(Zst, mean=0, sd=1)) pvalue <- 2\*(1-pnorm(Zst, mean=0, sd=1)) print(paste0("Z = ", Zst))

print(paste0("p value = ", pvalue))

# [1] "Z = 0.909334310849914"

# [1] "p value = 0.363173684651059"

# plot

normal\_area(mean=0, sd=1, lb=-Zst, ub=Zst, lwd=2, acolor=rgb(0, 0, 1, alpha=0.5))

text(-0.2, 0.12, paste0(100\*round(dtp,4), "%"), srt =90, cex=5)text(-2, 0.4, paste0("pvalue=", round(pvalue,4)), cex=2)

32

**Normal Distribution: One Sample Z-Test inR**

• Suppose the IQ in a certain population is normally distributed

# install.packages("BSDA")

# Z test for mean of one sample

with a mean of μ = 100 and standard deviation of σ = 15.

library(BSDA)

mu <- 100

• A scientist wants to know if a new medication affects IQ levels, so

sigma <- 15

# enter IQ levels for 20 patients

she recruits 20 patients to use it for one month and records their data <- c(88, 92, 94, 94, 96, 97, 97, 97, 99, 99,

IQ levels at the end of the month. 

105, 109, 109, 109, 110, 112, 112, 113, 114, 115)

# perform one sample z-test using z.test in BSDA packagez.test(data, mu=mu, sigma.x=sigma)

# perform one sample z-test manually Zst <- (mean(data)-mu)/(sigma/sqrt(length(data))) dtp <- 1 - 2\*(1-pnorm(Zst, mean=0, sd=1)) pvalue <- 2\*(1-pnorm(Zst, mean=0, sd=1)) print(paste0("Z = ", Zst))

print(paste0("p value = ", pvalue))

# [1] "Z = 0.909334310849914"

# [1] "p value = 0.363173684651059"

# plot

normal\_area(mean=0, sd=1, lb=-Zst, ub=Zst, lwd=2, acolor=rgb(0, 0, 1, alpha=0.5))

text(-0.2, 0.12, paste0(100\*round(dtp,4), "%"), srt =90, cex=5)text(-2, 0.4, paste0("pvalue=", round(pvalue,4)), cex=2)

# plot

normal\_area(mean=0, sd=1, lb=Zst, lwd=2, acolor=rgb(0, 0, 1, alpha=0.5))

text(1.2, 0.07, paste0(100\*round(pvalue/2,4), "%"), srt =90, cex=3)

text(-2, 0.4, paste0("pvalue=", round(pvalue,4)), cex=2) 33

**Normal Distribution: Two Sample Z-Test in R, exercise2**

• Suppose the IQ levels among individuals in two different cities are

library(BSDA)

known to be normally distributed each with population standard

#enter IQ levels for 20 individuals from each city

deviations of 15.

cityA = c(82, 84, 85, 89, 91, 91, 92, 94, 99, 99, 105, 109, 109, 109, 110, 112, 112, 113, 114, 114)

cityB = c(90, 91, 91, 91, 95, 95, 99, 99, 108, 109,

• A scientist wants to know if the mean IQ level between

109, 114, 115, 116, 117, 117, 128, 129, 130, 133)

individuals in city A and city B are different, so she selects a

#perform two sample z-test

z.test(x=cityA, y=cityB, mu=0, sigma.x=15, sigma.y=15)

simple random sample of 20 individuals from each city and

records their IQ levels. 

Two-sample z-Test

data: cityA and cityB

z = -1.7182, p-value = 0.08577

alternative hypothesis: true difference in means is not equal to095 percent confidence interval:

-17.446925 1.146925

sample estimates:

mean of x mean of y

100.65 108.80

**Exercise 2**: manually calculate z test for 2 samples usingR

https://www.statology.org/z-test-in-r/ 34

**Some Distribution Functions in R**

****

https://www.stat.umn.edu/geyer/old/5101/rlook.html 35

**Bayesian Classifier**

****36

**Two steps to create a ShinyApp**

1) Write R script to do the task and plot with basic, ggplot2 or plotly

2) Create a ShinyApp by modifying the template of App.R

2.1) The interface part 2.2) The Server part37

**Exercise: Modify Bayesian Decision theory App**

**Ex3:** Adding the slider to interactive change the meansofP(X|Y=female) and P(X|Y=male). Expain the observationforPosteriors

**Ex4:** Adding the slider to interactive change the variancesofP(X|Y=female) and P(X|Y=male). Expain the observationforPosteriors

38

**Exercise: create a ShinyApp for COVID-19 severity usingBayesian Classifier**

**Ex5:** Pridict serverity and mild of COVID-19 using Methylation Risk Score of COGRAM1) Write R script to do the task and plot all of Probability distribution below X = Methylation Risk Score of COGRAM

Y = Serverity or Mild

P(X) = estimate from data

P(X|Y=Serverity) = estimate from data

P(X|Y=Mild) = estimate from data

P(Y=Serverity) = 0.03

P(Y=Serverity|X) = ?

P(Y=Mild|X) = ?

2) Create a ShinyApp by modifying the template of App.R

2.1) The interface part 2.2) The Server part

**Ex6:** Pridict ICUD and non-ICUD of COVID-19 using Methylation Risk Score of ICUD Same as Ex1 but using the ICUD data

39

**Exercise: create a ShinyApp for optimzation of rate andepochin Gradient Descent**

2 + 4 − 1

**Ex7:** Minimize the function () = 7.1) Add slider for rate = 0.01, 0.1, 0.5, 1, 2, 10, 100 and keep n =10. Plotthetrajectory on the f(x) for the rate and explain the observation. 7.2) Add slider for rate = 0.01, 0.1, 0.5, 1, 2, 10, 100 and n = 10, 20, 50, 100, 500, 1000. Plot the trajectory on f(x) for rate and n. Thenexplaintheimportant observations.

2 + 4 + 2 − 1

**Ex8:** Minimize the function () = 7.1) Add slider for rate = 0.01, 0.1, 0.5, 1, 2, 10, 100 and keep n =10. Plotthetrajectory for the rate on the contour of f(x) and explain the observation. 7.2) Add slider for rate = 0.01, 0.1, 0.5, 1, 2, 10, 100 and n = 10, 20, 50, 100, 500, 1000. Plot the trajectory on the contour of f(x) for rate andn. Thenexplain the important observations.

40