Introduction to Data Science and Engineering

- Probability

Zhenqin (Michael) Wu / 吳楨欽

School of Computing and Data Science
University of Hong Kong

Slide deck originally created by RB Luo

BY NC SA

Some materials courtesy of Rafael A. Irizarry, and are modified from the original version.

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov

In this lecture

- **Probability and random numbers**
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov

What is Probability?

- Outcome of flipping a coin
- Chances being stuck in the MTR elevator queue
- If there is a storm signal No.8 next Monday
- Schrödinger's cat
  - *"God does not play dice"*

--- Albert Einstein

What is Probability?

- The chance that something will happen
- Given **infinite observations** of an event, the proportion of observations where a given outcome happens
- *"Probability is a mathematical language for quantifying uncertainty."* --- Larry Alan Wasserman

Case study: How to calculate π?

- Monte Carlo Simulation

Use Monte Carlo method to estimate π in R

```r
estimate_pi <- function(n_iter, seed=42){
  set.seed(seed)

  x <- runif(n_iter, min=0, max=1) - 0.5
  y <- runif(n_iter, min=0, max=1) - 0.5
  dist <- sqrt(x**2 + y**2)
  area <- mean(dist <= 0.5)

  pi <- 4 * area
  pi
}
```

- `runif` is not run-if, it is r-unif
  - Sampling from a uniform distribution between `min` and `max`
- `set.seed` for setting the random seed

```r
> n_iters <- c(1e1, 1e2, 1e3, 1e4, 1e5)
> seeds <- seq(1:100)
> df <- expand.grid(n_iters, seeds) |> setNames(c("n_iter", "seed"))
> df$pi_estimates <- mapply(estimate_pi, df$n_iter, df$seed)
```

Use Monte Carlo method to estimate π in R

```
> df |>
+   group_by(n_iter) |>
+   summarize(mean=mean(pi_estimates), sd=sd(pi_estimates)) |>
+   mutate(mean=format(mean, digits=5))
# A tibble: 5 × 3
  n_iter mean        sd
   <dbl> <chr>     <dbl>
1     10 3.1560 0.533
2    100 3.1336 0.180
3   1000 3.1349 0.0518
4  10000 3.1415 0.0173
5 100000 3.1410 0.00481
```

```
> df |>
+   ggplot(aes(x = n_iter, y = pi_estimates, group = n_iter)) +
+   geom_hline(yintercept = pi, linetype = "dashed") +
+   geom_boxplot() +
+   scale_x_log10() +
+   theme_minimal()
```

Random Number Generator (RNG)

```
> set.seed(42)
> round(runif(10,1,10))
 [1] 9 9 4 8 7 6 8 2 7 7
> round(runif(10,1,10))
 [1]  5  7  9  3  5  9 10  2  5  6
> set.seed(42)
> round(runif(10,1,10))
 [1] 9 9 4 8 7 6 8 2 7 7
> round(runif(10,1,10))
 [1]  5  7  9  3  5  9 10  2  5  6
> set.seed(21)
> round(runif(10,1,10))
 [1]  8  3  7  3 10  9  2  3 10  9
> round(runif(10,1,10))
 [1] 7 9 2 7 3 1 6 4 5 7
> set.seed(42)
> round(runif(10,1,10))
 [1] 9 9 4 8 7 6 8 2 7 7
> round(runif(10,1,10))
 [1]  5  7  9  3  5  9 10  2  5  6
```

- After setting a seed, the random number output sequence is deterministic
- The same seed always gives you the same sequence
- A different seed gives a different sequence
- **!! Use the same seed if the results requires to be repeatable !!**
- This is known as pseudo RNG

How to get true random number?
Pick a random number between 0-100

Pick a random number between 0-100

- Top picks
  - 73
  - 77
  - 37
  - 69
  - 42 (the answer to everything)
  - 7

The Hitchhiker's Guide to the Galaxy, Douglas Adams
Veritasium: https://www.youtube.com/watch?v=d6iQrh2TK98&t=907s

PseudoRandom Number Generator

- One example: linear congruential generator

The generator is defined by the recurrence relation:

$$X_{n+1} = (aX_n + c) \mod m$$

where X is the sequence of pseudo-random values, and

m, $0 < m$ — the "modulus"

a, $0 < a < m$ — the "multiplier"

c, $0 \leq c < m$ — the "increment"

X_0, $0 \leq X_0 < m$ — the "seed" or "start value"

https://en.wikipedia.org/wiki/Linear_congruential_generator

(True) Random Number Generator

- Need to base on actual physical process:
  - thermal noises, signal jitter, Brownian motion, quantum phenomena, nuclear decay, etc.
- Rate-limited, cannot generate random numbers at high throughput. Why?

Article | Open access | Published: 09 August 2021

Quantum generators of random numbers

Marcin M. Jacak, Piotr Jóźwiak, Jakub Niemczuk & Janusz E. Jacak

Scientific Reports **11**, Article number: 16108 (2021) | Cite this article

28k Accesses | **65** Citations | **30** Altmetric | Metrics

a random bit sequence with velocity of 1 Mb/s

Jacak, Marcin M., et al. "Quantum generators of random numbers." *Scientific Reports* 11.1 (2021): 16108.

In this lecture

- Probability and random numbers
- **Discrete probability**
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov

Discrete Probability

- Outcomes when using Monte Carlo simulation to calculate π:
  - The point is within the circle
    - Probability is [area of the circle]/[area of the square envelope]
  - The point is not in the circle
    - Probability is 1 - [area of the circle]/[area of the square envelope]

Discrete Probability

- 5 beads in a pot:

```
> beads <- rep(c("red", "blue"), times = c(2,3))
> beads
[1] "red"  "red"  "blue" "blue" "blue"
```

- Pick a bead:

```
> sample(beads, 1)
[1] "blue"
```

```
> mean(replicate(1e5, sample(beads, 1) == "blue"))
[1] 0.60048
```

Notations

- Ω: Sample Space, all possible outcomes of a random experiment
- A: Event ($\mathrm{A} \subseteq \Omega$), one (or multiple) possible outcome(s)
- $P(\mathrm{A})$: Probability of event A
- Properties:
  - $P(\Omega) = 1$ (Something will happen)
  - $P(A) \geq 0$, for any A (Every outcome is possible)
  - If $\mathrm{A}_1, \mathrm{A}_2, \ldots$ are different and ***<u>independent</u>*** events then

$$P\left(\bigcup_i \mathrm{A}_i\right) = \sum_i P(\mathrm{A}_i)$$

In this lecture

- Probability and random numbers
- Discrete probability
- **Independence and conditional probability**
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov

Independence

- We say two events are independent if the outcome of one does not affect the other
- Are the events independent in
  - Outcome of flipping a coin (two sequential trials)
  - Chances being stuck in the MTR elevator queue (Monday / Tuesday)
  - If there is a storm signal No.8 (next Monday / next Tuesday)
  - Keep unfolding cards from a deck, the chance of getting an Ace
  - Schrödinger’s (two) cat(s)

Other probability properties

- For two events
  - if $\mathrm{B} \subseteq \mathrm{A}$ then $P(\mathrm{A}) \geq P(\mathrm{B})$
  - $P(\neg \mathrm{A}) = 1 - P(\mathrm{A})$
  - $P(\mathrm{A} \cup \mathrm{B}) \leq P(\mathrm{A}) + P(\mathrm{B})$
    - Two events are independent if and only if (iff) $P(\mathrm{A} \cup \mathrm{B}) = P(\mathrm{A}) + P(\mathrm{B})$
  - $P(\mathrm{A}, \mathrm{B}) \leq \min(P(\mathrm{A}), P(\mathrm{B}))$
    - Two events are independent iff $P(\mathrm{A}, \mathrm{B}) = P(\mathrm{A}) * P(\mathrm{B})$

Independence

- For the five-bead example, are the first draw and the second draw independent?

```
> p <- mean(replicate(1e5, all(sample(beads, 1) == "blue")))
> p ** 2
[1] 0.36142
```

```
> mean(replicate(1e5, all(sample(beads, 2) == "blue")))
[1] 0.2986
```

Independence

- Are the events independent in
  - Outcome of flipping two coins;
  - Chances being stuck in the MTR elevator queue (Monday / Tuesday);
  - If there is a storm signal No.8 (next Monday / next Tuesday);
  - Keep unfolding cards from a deck, the chance of getting an Ace;
  - Schrödinger’s cat(s) in two experiments on the same day.

Independence

- Are the events independent in
  - Outcome of flipping two coins;
  - Chances being stuck in the MTR elevator queue (Monday / Tuesday);
  - If there is a storm signal No.8 (next Monday / next Tuesday);
  - Keep unfolding cards from a deck, the chance of getting an Ace;
  - Schrödinger's cat(s) in two experiments on the same day.
- Some common mistakes:
  - I have lost 10 times in a random game (flipping coin), I will win next time.
  - Results from an online survey including 1M participants are unbiased.
  - The 2008 financial crisis: more on that later

Conditional probabilities

- When events are not independent, conditional probabilities are useful
  - $P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{A},\mathrm{B})}{P(\mathrm{B})}$
  - $P(\text{ Getting an Ace } \mid \text{ Four Ace on the table already }) = 0$

Event A: mentioning ”happy” in a message $P(\mathrm{A}) = 0.01$
Event B: mentioning “birthday” in a message $P(\mathrm{B}) = 0.001$

$$P(A, B) = 0.0005$$

$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{A}, \mathrm{B})}{P(\mathrm{B})} = 0.5$$

Reconciliating independence and conditional probability

Event B: ten consecutive coin flips are all heads $P(\mathrm{B}) = 0.00097656$
Event A: next coin flip is still head $P(\mathrm{A}) = 0.5$

$$P(A, B) = 0.00048828$$
$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{A}, \mathrm{B})}{P(\mathrm{B})} = 0.5 = P(\mathrm{A})$$

Two events, *A* and *B*, are independent iff $P(\mathrm{A}, \mathrm{B}) = P(\mathrm{A}) * P(\mathrm{B})$

In other words, independence exists iff $P(\mathrm{A}|\mathrm{B}) \ = \ P(\mathrm{A})$
i.e., Observing B has no effect on probability of A

What’s underlying LLMs

Improving Language Understanding by Generative Pre-Training

Alec Radford
OpenAI
alec@openai.com

Karthik Narasimhan
OpenAI
karthikn@openai.com

Tim Salimans
OpenAI
tim@openai.com

Ilya Sutskever
OpenAI
ilyasu@openai.com

Given an unsupervised corpus of tokens $\mathcal{U} = \{u_1, \ldots, u_n\}$, we use a standard language modeling objective to maximize the following likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta) \quad (1)$$

Radford, Alec, et al. "Improving language understanding by generative pre-training." (2018): 3.

Calculating combinations and permutations

- What is the probability that if I draw five cards without replacement, I get all cards of the same suit, what is known as a "flush" in poker?

- To generate a deck of cards:

```
> numbers <- c(
+     "Ace", "Deuce", "Three", "Four", "Five", "Six", "Seven",
+     "Eight", "Nine", "Ten", "Jack", "Queen", "King")
> deck <- expand.grid(number=numbers, suit=suits)
> deck <- paste(deck$number, deck$suit)
> sample(deck, 5, replace=FALSE)
[1] "King Diamonds"  "Three Spades"   "Eight Clubs"    "Queen Diamonds" "Ace Hearts"
```

Case study: getting a flush

Permutation:

- 3,1 is different than 1,3
- (1,1), (2,2), and (3,3) do not appear

```
> library(gtools)
> permutations(3, 2)
     [,1] [,2]
[1,]    1    2
[2,]    1    3
[3,]    2    1
[4,]    2    3
[5,]    3    1
[6,]    3    2
```

- `permutations` from `gtools` provide an interface to enumerate all possible permutations (of a certain size) of a vector

```
> three_card_combinations <- permutations(52, 3, v=deck)
> dim(three_card_combinations)
[1] 132600      3
> three_card_combinations[sample(dim(three_card_combinations)[[1]], 5),]
     [,1]             [,2]           [,3]
[1,] "Three Spades"   "Queen Clubs"  "Three Clubs"
[2,] "Ten Clubs"      "Five Spades"  "Ace Hearts"
[3,] "Ten Clubs"      "Four Hearts"  "Deuce Diamonds"
[4,] "Jack Diamonds"  "Ace Clubs"    "Nine Clubs"
[5,] "Eight Diamonds" "Seven Hearts" "Three Clubs"
```

- `permutations(52,5)` will have 300M+ entries, we will work on `permutations(52,3)` instead.

Case study: getting a flush

```
> heart_cards <- paste(numbers, "Hearts")
> p_first_is_heart <- mean(three_card_combinations[,1] %in% heart_cards)
> print(p_first_is_heart)
[1] 0.25
> p_second_is_heart <- mean(three_card_combinations[,2] %in% heart_cards)
> print(p_second_is_heart)
[1] 0.25
> p_third_is_heart <- mean(three_card_combinations[,3] %in% heart_cards)
> print(p_third_is_heart)
[1] 0.25
> p_all_are_heart <- mean(
+   three_card_combinations[,1] %in% heart_cards &
+   three_card_combinations[,2] %in% heart_cards &
+   three_card_combinations[,3] %in% heart_cards)
> print(p_all_are_heart)
[1] 0.012941
```

- Probability of all hearts is 0.0129 < 0.0156 = 0.25 * 0.25 * 0.25

Case study: getting a flush

- Event A: first card is heart
- Event B: second card is heart
- Event C: third card is heart
- $P(\mathrm{A}) = P(\mathrm{B}) = P(\mathrm{C}) = 0.25$

- What is the probability of first two cards are both hears:

$$P(\mathrm{A}, \mathrm{B}) = P(\mathrm{A})P(\mathrm{B}|\mathrm{A})$$

$$P(\mathrm{A}) = \frac{13}{52} = 0.25$$

$$P(\mathrm{B}|\mathrm{A}) = \frac{13-1}{52-1} = 0.2353$$

Case study: getting a flush

```
> three_card_combinations |>
+   as.data.frame() |>
+   summarize(first_is_heart = mean(V1 %in% heart_cards))
  first_is_heart
1           0.25
> three_card_combinations |>
+   as.data.frame() |>
+   filter(V1 %in% heart_cards) |>
+   summarize(second_is_heart = mean(V2 %in% heart_cards))
  second_is_heart
1       0.2352941
> three_card_combinations |>
+   as.data.frame() |>
+   filter(V1 %in% heart_cards & V2 %in% heart_cards) |>
+   summarize(second_is_heart = mean(V3 %in% heart_cards))
  second_is_heart
1            0.22
```

- Event A: first card is heart
- Event B: second card is heart
- Event C: third card is heart

$$P(\mathrm{A}) = \frac{13}{52} = 0.25$$

$$P(\mathrm{B}|\mathrm{A}) = \frac{13-1}{52-1} = 0.2353$$

$$P(\mathrm{C}|\mathrm{A},\mathrm{B}) = \frac{13-2}{52-2} = 0.22$$

$$P(\mathrm{A},\mathrm{B},\mathrm{C}) = 0.25 * 0.2353 * 0.22 = 0.0129$$

Case study: getting a flush

```r
is_flush <- function(seed=NA){
  if (!is.na(seed)){
    set.seed(seed)
  }

  hand <- sample(deck, 5, replace=FALSE)
  suits_in_hand <- str_split(hand, "\\s", simplify = TRUE)
  length(unique(suits_in_hand[,2])) == 1
}

set.seed(42)
mean(replicate(1e5, is_flush()))
```

```
> mean(replicate(1e5, is_flush()))
[1] 0.00198
```

```
> set.seed(35)
> hand <- sample(deck, 5, replace=FALSE)
> hand
[1] "Ten Diamonds"   "Six Diamonds"   "Eight Diamonds" "Ace Diamonds"   "Seven Diamonds"
```

- Probability of a "flush":

$$P(\text{flush}) = \frac{13}{52} * \frac{12}{51} * \frac{11}{50} * \frac{10}{49} * \frac{9}{48} * 4$$
$$= 0.00198$$

Case study: getting a natural 21

```
> combinations(3,2)
     [,1] [,2]
[1,]    1    2
[2,]    1    3
[3,]    2    3
```

- In Blackjack, if you get an Ace and a face card in the first draw, it is called a natural 21 and you win automatically.
- The order of the two cards doesn't matter. In such cases, we look at `combinations` instead of `permutations`

```
> aces <- paste("Ace", suits)
>
> facecard <- c("King", "Queen", "Jack", "Ten")
> facecard <- expand.grid(number = facecard, suit = suits)
> facecard <- paste(facecard$number, facecard$suit)
>
> hands <- combinations(52, 2, v = deck)
> mean((hands[,1] %in% aces & hands[,2] %in% facecard) |
+      (hands[,2] %in% aces & hands[,1] %in% facecard))
[1] 0.04826546
```

Case study: getting a natural 21

```r
is_21 <- function(seed=NA){
  if (!is.na(seed)){
    set.seed(seed)
  }

  hand <- sample(deck, 2, replace=FALSE)
  flag1 <- hand[[1]] %in% aces & hand[[2]] %in% facecard
  flag2 <- hand[[2]] %in% aces & hand[[1]] %in% facecard
  flag1 | flag2
}

set.seed(42)
mean(replicate(1e6, is_21()))
```

```
> mean(replicate(1e6, is_21()))
[1] 0.048031
```

- Probability of a "natural 21":

$$P = \frac{4}{52} * \frac{16}{51} * 2 = 0.0483$$

Other well-known examples

- The Monty Hall problem
- The birthday paradox
  - In a group of 50 people, there’s a 96%+ chance that two people share the same birthday.

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- **Continuous probability: expected value and variance**
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov

Continuous probability

- Continuous probability focused on analyzing the distributions and observations of continuous variables

- What is the proportion of students in HKU that have heights of 170cm?
- What is the proportion of students in HKU that have heights of 170.000000000cm?

Continuous probability properties

- We call X is a continuous random variable if it can take on an infinite number of values between any two given values
- X: observation $X(\omega)$ on inches of snowfall in a snowstorm event ω
- What is the probability we receive (at least) **a** inches?
  - $P(X \geq \boldsymbol{a}) := P(\{\omega: X(\omega) \geq \boldsymbol{a}\})$
- What is the probability we receive between **a** and **b** inches?
  - $P(\boldsymbol{a} \leq X \leq \boldsymbol{b}) := P(\{\omega: \boldsymbol{a} \leq X(\omega) \leq \boldsymbol{b}\})$
- Probability of receiving exactly i inches of snowfall is zero
  - $P(X = i) := 0$, for all $i \in \Omega$

Cumulative distribution functions (CDF)

```
> data(heights)
> height_cdf <- function(threshold){
+   heights |> summarize(value = mean(height <= threshold)) |> pull(value)
+ }
> height_cdf(170 / 2.53)
[1] 0.3885714
```

```
df <- data.frame(val = seq(120, 220, 1))
df <- df |>
  mutate(val_in_inch = val / 2.53,
         ecdf = sapply(val_in_inch, height_cdf))
df |> ggplot(aes(val, ecdf)) +
  geom_line() +
  xlab("Height (cm)") +
  ylab("empirical CDF of student height") +
  theme_minimal()
```

- Empirical CDF/eCDF(a): given all the **observations**, proportion of observations whose values are smaller than (or equal to) a

$$F(a) = \Pr(X \leq a)$$

Cumulative distribution functions (CDF)

```r
# Plot CDF of Gaussian
df <- data.frame(
  val = seq(-5, 5, 0.01),
  cdf = pnorm(seq(-5, 5, 0.01), 0, 1)
)
df |> ggplot(aes(val, cdf)) +
  geom_line() +
  ylab("CDF of a standard normal\n(mean=0, sd=1)") +
  theme_minimal()
```

- $\mathrm{CDF}(a)$: given the **distribution**, probability of a continuous random variable that takes values smaller than (or equal to) a

$$F(a) = \Pr(X \leq a)$$

- Empirical CDF/$\mathrm{eCDF}(a)$: given all the **observations**, proportion of observations whose values are smaller than (or equal to) a

Probability density functions (PDF)

- Once a CDF is defined, we can use it to compute the probability of any subsets or intervals of values.

$$\Pr(a < X \leq b) = F(b) - F(a)$$

- When a is equal to b, then probability goes to 0.

- Define $\mathrm{PDF}(a)$ as the first derivative (gradient) of CDF at a; CDF can be constructed by the integral of PDF

$$F(b) - F(a) = \int_a^b f(x)\,dx$$

```r
df <- data.frame(
  val = seq(-5, 5, 0.01),
  pdf = dnorm(seq(-5, 5, 0.01), 0, 1)
)
df |> ggplot(aes(val, pdf)) +
  geom_line() +
  ylab("PDF of a standard normal\n(mean=0, sd=1)") +
  theme_minimal()
```

Probability density functions (PDF)

- Is there empirical PDF?

Probability density functions (PDF)

- Is there empirical PDF?

```
height_summary <- heights |>
  filter(sex == "Male") |>
  summarize(mean=mean(height), sd=sd(height))

heights |>
  filter(sex == "Male") |>
  ggplot(aes(height)) +
  geom_density(fill="blue", alpha=0.2) +
  stat_function(
    fun=dnorm,
    args=list(mean=height_summary$mean, sd=height_summary$sd),
    color="red") +
  ylab("Density") +
  theme_minimal()
```

Use Monte Carlo simulations for continuous variables

The Normal Distribution

Description

Density, distribution function, quantile function and random generation for the normal distribution with mean equal to `mean` and standard deviation equal to `sd`.

Usage

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

```
> sampled_heights <- data.frame(
+   height = rnorm(500, mean=height_summary$mean, sd=height_summary$sd))
> sampled_heights |> ggplot(aes(height)) + geom_histogram() + theme_minimal()
```

Use Monte Carlo simulations for continuous variables

```
> top_out_of_1000 <- function(){
+   sampled_heights <- rnorm(1000, mean=height_summary$mean, sd=height_summary$sd)
+   max(sampled_heights)
+ }
> top_heights <- data.frame(height = replicate(1e5, top_out_of_1000()))
> top_heights |> ggplot(aes(height)) + geom_histogram() + theme_minimal()
```

- This is not a normal distribution
- These values are modeled through Gumbel distribution

https://en.wikipedia.org/wiki/Gumbel_distribution

Learn more about different probability distributions in STAT2601/2

In probability theory and statistics, there are several relationships among probability distributions. These relations can be categorized in the following groups:

- One distribution is a special case of another with a broader parameter space
- Transforms (function of a random variable);
- Combinations (function of several variables);
- Approximation (limit) relationships;
- Compound relationships (useful for Bayesian inference);
- Duality;
- Conjugate priors.

Working with continuous distributions in R

The Normal Distribution

Description

Density, distribution function, quantile function and random generation for the normal distribution with mean equal to `mean` and standard deviation equal to `sd`.

Usage

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

The Student t Distribution

Description

Density, distribution function, quantile function and random generation for the t distribution with `df` degrees of freedom (and optional non-centrality parameter `ncp`).

Usage

```
dt(x, df, ncp, log = FALSE)
pt(q, df, ncp, lower.tail = TRUE, log.p = FALSE)
qt(p, df, ncp, lower.tail = TRUE, log.p = FALSE)
rt(n, df, ncp)
```

The (non-central) Chi-Squared Distribution

Description

Density, distribution function, quantile function and random generation for the chi-squared (χ^2) distribution with `df` degrees of freedom and optional non-centrality parameter `ncp`.

Usage

```
dchisq(x, df, ncp = 0, log = FALSE)
pchisq(q, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)
qchisq(p, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)
rchisq(n, df, ncp = 0)
```

The Exponential Distribution

Description

Density, distribution function, quantile function and random generation for the exponential distribution with rate `rate` (i.e., mean `1/rate`).

Usage

```
dexp(x, rate = 1, log = FALSE)
pexp(q, rate = 1, lower.tail = TRUE, log.p = FALSE)
qexp(p, rate = 1, lower.tail = TRUE, log.p = FALSE)
rexp(n, rate = 1)
```

Expected value and variance

Two important properties for a random variable:

- Expected value: $\mathrm{E}(X)$
  - Coin toss: head = 1, tail = -1; expected value is 0 in a perfect 50/50 scenario
  - Investment: weighted average of each possible outcome by its probability; expected value is the mean gain over many repetitions

- Variance: $\mathrm{Var}(X) = \mathrm{E}[(X - \mathrm{E}(X))^2]$ and standard deviation $\mathrm{SD}(X) = \sqrt{\mathrm{Var}(X)}$
  - Investment: chance of getting high return or loss; measurement of risk
  - Tolerance (engineering): manufacture of the same item may have tiny variability in the end product; low tolerance = high quality

How to calculate them

Say random variable X may take values $x_1, x_2, \ldots$

$$\mathrm{E}(X) = \sum x_i P(x_i)$$

In real-world data, observations directly came from $P(X)$, the average of observations $\bar{X}$ converges to $\mathrm{E}(X)$ as number of observations increase (the Law of Large Numbers).

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- **The Law of Large Numbers, Central Limit Theory**
- Case study: bank loan
- The debate between Markov and Nekrasov

The Law of Large Numbers

X: getting a natural 21 = 1, else = 0

$\mathrm{E}(X) = 0.0483$

```r
df <- data.frame(repetitions = round(10**seq(1, 6, 0.5)))
run_21_n_times <- function(n=10){
  outcome <- replicate(n, is_21())
  mean(outcome)
}
df <- df |> mutate(chance = sapply(repetitions, run_21_n_times))
df |>
  ggplot(aes(repetitions, chance)) +
  geom_line() +
  geom_point() +
  scale_x_log10() +
  geom_hline(yintercept=0.0483, lty="dashed") +
  theme_minimal()
```

The Law of Large Numbers

X: sample a value from Gaussian distribution (mean = 0)

$\mathrm{E}(X) = 0$

```r
df <- data.frame(repetitions = round(10**seq(1, 7, 0.5)))
sample_standard_normal <- function(n=10, sd=1){
  outcome <- rnorm(n, mean = 0, sd = sd)
  mean(outcome)
}
df <- df |> mutate(chance = sapply(repetitions, sample_standard_normal))
df <- df |> mutate(chance_sd10 = sapply(repetitions, sample_standard_normal, sd=10))
df |>
  ggplot() +
  geom_line(aes(repetitions, chance)) +
  geom_point(aes(repetitions, chance)) +
  geom_line(aes(repetitions, chance_sd10), color="red") +
  geom_point(aes(repetitions, chance_sd10), color="red") +
  scale_x_log10() +
  geom_hline(yintercept=0., lty="dashed") +
  theme_minimal()
```

How to calculate them

Say random variable X may take values $x_1, x_2, \ldots$

$\mathrm{E}(X) = \sum x_i P(x_i)$ can be approximated by $\bar{X}$

Similarly, $\mathrm{SD}(X) = \sqrt{\sum (x_i - \mathrm{E}(X))^2 P(x_i)}$ can be approximated by:

$$\frac{1}{n} \sum (X_{obs} - \bar{X})^2$$

Because observations directly came from $P(X)$

- Divide by n or $n - 1$?
- For this class, we will use n if not otherwise specified

Some important properties

- For n i.i.d. samples of X:

$$\mathrm{E}(X_1 + X_2 + \cdots + X_n) = \mathrm{E}(X_1) + \mathrm{E}(X_2) + \cdots + \mathrm{E}(X_n)$$

$$E\left(\frac{X_1 + X_2 + \cdots + X_n}{n}\right) = \frac{\mathrm{E}(X_1) + \mathrm{E}(X_2) + \cdots + \mathrm{E}(X_n)}{n} = E(X_1)$$

$$\mathrm{Var}(X_1 + X_2 + \cdots + X_n) = \mathrm{Var}(X_1) + \mathrm{Var}(X_2) + \cdots + \mathrm{Var}(X_n)$$

$$Var(CX_1) = Var(X_1) * C^2$$

$$\mathrm{Var}\left(\frac{X_1 + X_2 + \cdots + X_n}{n}\right) = \frac{\mathrm{Var}(X_1) * n}{n^2} = \frac{\mathrm{Var}(X_1)}{n}$$

$$\mathrm{SD}\left(\frac{X_1 + X_2 + \cdots + X_n}{n}\right) = \frac{\mathrm{SD}(X_1)}{\sqrt{n}}$$

(this is often referred to as the standard error of sample mean)

Central Limit Theorem (CLT)

For random variable X that follows **ANY** distribution (with finite mean and variance), **sample mean** of n i.i.d. observations (this is also a random variable) will converge to:

a Gaussian distribution with $\mathrm{E}(X)$ as mean, and $\frac{\mathrm{SD}(X)}{\sqrt{n}}$ as sd, when $\boldsymbol{n}$ **is large**.

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- **Case study: bank loan**
- The debate between Markov and Nekrasov

Case study: bank loan

Suppose you run a small bank that provides loan to customers that can be trusted to make payments.

- Historically, in a given year, only 2% of your customers default (fail to pay back the loan).
- By charging everybody just a bit extra in interest, you can make up the losses incurred due to that 2% and also cover your operating costs.
- If you set the interest rates too high, your clients will go to another bank.

What interest rate should you charge?

Case study: bank loan

```r
# Case study: loan
principal_per_loan <- 180000
operational_loss <- 20000
loss_per_foreclosure <- -(principal_per_loan + operational_loss)

sim_average_return <- function(interest_rate=0.02, default_rate=0.02, loans=1000){
  defaults <- sample(
      c(0,1), loans, prob=c(1-default_rate, default_rate), replace = TRUE)
  balance <- sum(defaults * loss_per_foreclosure) +
      sum((1 - defaults) * principal_per_loan * interest_rate)
  balance / loans
}
```

1,000 loans of $180k. After adding up all costs, suppose your bank loses $200,000 per foreclosure.

Assume that default rate is 2%:

- If defaulted, the bank will lose $200,000
- If not, the bank will earn $180k x interest rate

```
> set.seed(42)
> sim_average_return(0.02, 0.02)
[1] 749.6
> set.seed(43)
> sim_average_return(0.02, 0.02)
[1] -1286.4
```

Outcome of each loan is a random variable; `sim_average_return` is calculating its sample mean, which is also a random variable

```r
return_sampled <- data.frame(
  return=replicate(1e5, sim_average_return(0.02, 0.02))
)
return_sampled |>
  ggplot(aes(return)) +
  geom_density(adjust = 1.5, fill = "blue", alpha = 0.2) +
  geom_vline(xintercept = 0) +
  theme_minimal()
```

We can use Monte Carlo simulation to get an idea on the distribution of the average return

Takeaway:

- Sample mean (average return) is very “normal” (CLT);
- Majority of the returns are negative

```r
return_sampled <- data.frame(
  interest_rate=rep(seq(0.02, 0.04, 0.001), 1e5))

return_sampled <- return_sampled |>
  mutate(return=sapply(interest_rate, sim_average_return))

return_sampled |>
  mutate(interest_rate=as.factor(interest_rate)) |>
  ggplot(aes(return, interest_rate)) +
    geom_density_ridges(bandwidth=300, fill="blue", alpha=0.2) +
    geom_vline(xintercept=0., lty="dashed") +
    theme_minimal()
```

Banks will, therefore, increase the interest rate.

When interest rate surpasses 2.3%, the center of the distribution shifts to positive.

```r
return_sampled |>
  filter(interest_rate==0.023) |>
  ggplot(aes(return)) +
    stat_ecdf(geom = "step", color = "black") +
    geom_vline(xintercept=0., lty="dashed") +
    theme_minimal() +
    xlab("return (interest rate = 2.3%)") +
    ylab("eCDF")
```

This is still not enough! We need an interest rate that lowers the chance of losing money to lower than 1%.

If interest rate = 2.3%, the bank has a ~50% chance of losing money

```r
return_sampled |>
  group_by(interest_rate) |>
  summarize(loss_prob = mean(return < 0)) |>
  ggplot(aes(interest_rate, loss_prob)) +
    geom_point() +
    geom_line() +
    geom_hline(yintercept=0.01, lty="dashed") +
    theme_minimal()
```

One option is to keep increasing the interest rate.

At interest rate=3.5%, the chance of losing money is ~1%

```r
return_sampled <- data.frame(
  loans=rep(seq(1000, 5000, 500), 1e4))

f <- function(loans){
  sim_average_return(interest_rate=0.03, default_rate=0.02, loans=loans)
}

return_sampled <- return_sampled |>
  mutate(return=sapply(loans, f))

return_sampled |>
  group_by(loans) |>
  summarize(loss_prob = mean(return < 0)) |>
  ggplot(aes(loans, loss_prob)) +
  geom_point() +
  geom_line() +
  geom_hline(yintercept = 0.01, lty = "dashed") +
  theme_minimal()
```

Another option is to increase the size of loans (why?)

At interest rate=3%, giving out 3000 loans (instead of 1000) reduces losing chance to <1%

- Return is a sample mean, whose standard error will decrease if n increases.

```r
return_sampled <- data.frame(
  interest_rate=rep(seq(0.03, 0.1, 0.005), 1e5))

f <- function(i){
  sim_average_return(interest_rate=i, default_rate=0.04, loans=1000)
}

return_sampled <- return_sampled |>
  mutate(return=sapply(interest_rate, f))

return_sampled |>
  group_by(interest_rate) |>
  summarize(loss_prob = mean(return < 0)) |>
  ggplot(aes(interest_rate, loss_prob)) +
  geom_point() +
  geom_line() +
  geom_hline(yintercept = 0.01, lty = "dashed") +
  theme_minimal()
```

In economy recession, default rate might be higher than 2%;

If default rate is at 4%, when giving out 1000 loans, the bank needs to have a 6.5+% interest rate to avoid losing

So far, we have assumed that the events of defaults are **independent** across loans, but is that true?

In economic crisis, everyone is affected, everyone will have a higher chance of bankruptcy.

```r
sample_return_when_default_rate_fluc <- function(){
  default_rate_fluc <- runif(1, -0.02, 0.02)
  default_rate <- 0.02 + default_rate_fluc
  balances <- sim_average_return(
      interest_rate = 0.03, default_rate = default_rate, loans = 3000)
  balances
}

returns_with_fluc <- replicate(1e5, sample_return_when_default_rate_fluc())
returns_flat <- data.frame(return = returns_with_fluc)

returns_flat |>
  ggplot(aes(return)) +
    geom_density(fill="blue", alpha=0.2) +
    geom_vline(xintercept=0., lty="dashed") +
    xlab("return (interest rate = 3%, 3000 loans)") +
    theme_minimal()
```

Loss rate < 1% if default rate is fixed

Loss rate ~ 34%

```r
sample_return_when_default_rate_fluc <- function(){
  default_rate_fluc <- runif(1, -0.02, 0.02)
  default_rate <- 0.02 + default_rate_fluc
  balances <- sim_average_return(
      interest_rate = 0.03, default_rate = default_rate, loans = 3000)
  balances
}

returns_with_fluc <- replicate(1e5, sample_return_when_default_rate_fluc())
returns_flat <- data.frame(return = returns_with_fluc)

returns_flat |>
  ggplot(aes(return)) +
    geom_density(fill="blue", alpha=0.2) +
    geom_vline(xintercept=0., lty="dashed") +
    xlab("return (interest rate = 3%, 3000 loans)") +
    theme_minimal()
```

- This is not normal?
- Does it satisfy CLT?

More factors

- Higher interest rate leads to higher default rate

For example: a 10M HKD property paid over 30 years

@2.3% interest rate
30,784 HKD monthly repayment

@7% interest rate
53,224 HKD monthly repayment

More factors

- Macro-economic changes & Black swan events
- Adjustable-rate mortgages
- Complex financial products that hide risks
- Cascading defaults

Clips from the 2015 movie “The Big Short”

- https://www.youtube.com/watch?v=epb98OcFLZE
- https://www.youtube.com/watch?v=xbiDrzTd8fE
- https://www.youtube.com/watch?v=A25EUhZGBws

Some explanations:

- https://www.youtube.com/watch?v=zZZb8GhsEfc

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- **The debate between Markov and Nekrasov**

The debate between Markov and Nekrasov

For random variable X that follows **ANY** distribution (with finite mean and variance), **sample mean** of n **i.i.d.** observations (this is also a random variable) will converge to:

a Gaussian distribution with $\mathrm{E}(X)$ as mean, and $\frac{\mathrm{SD}(X)}{\sqrt{n}}$ as sd, when $\boldsymbol{n}$ **is large**.

i.i.d.: independent and identically distributed, this is usually a very strong assumption

Is it necessary? Can we get convergence (LLN) and normality (CLT) without i.i.d.?

Nekrasov's view

- Independence is a NECESSARY condition for LLN/CLT

- If you observed normal distribution, then the underlying data generation process is independent
  - If exam score distribution converges to a normal, then every participating student's mid-term performance is independent
  - Have you asked your classmates about midterm preparation?

Markov's counter example

- In a free-form text, every letter is dependent on its previous letter(s).
- Analysis on "Eugene Onegin" by Alexander Pushkin

He was too young to have been blighted
by the cold world's corrupt finesse;
his soul still blossomed out, and lighted
at a friend's word, a girl's caress.
In heart's affairs, a sweet beginner,
he fed on hope's deceptive dinner;
the world's éclat, its thunder-roll,
still captivated his young soul.
He sweetened up with fancy's icing
the uncertainties within his heart;
for him, the objective on life's chart
was still mysterious and enticing—
something to rack his brains about,
suspecting wonders would come out.

- Even the observations are all dependent, they still follow LLN.
- *Free will (independence) is not necessary to do probability*

Source: https://www.americanscientist.org/article/first-links-in-the-markov-chain

Markov chain

Source: https://www.americanscientist.org/article/first-links-in-the-markov-chain

Markov chain

- Monte Carlo simulation
- PageRank (Google)
- Hidden Markov Models for DNA/protein sequence analysis
- Markov Decision Processes
- Language models
  - Model how the current letter is dependent on its previous letters (context window)
- Many, many more

In this lecture

- Probability and random numbers
- Discrete probability
- Independence and conditional probability
- Continuous probability: expected value and variance
- The Law of Large Numbers, Central Limit Theory
- Case study: bank loan
- The debate between Markov and Nekrasov