AMS 597: Statistical Computing

Pei-Fen Kuan (c)

Applied Math and Stats, Stony Brook University

Random sampling

- The basic notion of a random sample is to deal from a well-shuffled pack of cards or picking numbered balls from a well-stirred urn.
- In R, you can simulate these situations with the sample function. If you want to pick five numbers at random from the set 1:40, then you can write.

```
sample(1:40,5)
sample(c("H","T"), 10, replace=T)
sample(c("succ", "fail"), 10, replace=T, prob=c(0.9, 0.1))
```

Probability calculations and combinatorics

- In R, the choose function can be used to calculate the number of ways to choose 2 numbers out of 5

```r
choose(5,2)
```

```
## [1] 10
```

- You can also use the comb function to generate all combinations of n elements, taken m at a time

```r
combn(5,2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]    1    1    1    1    2    2    2    3    3     4
## [2,]    2    3    4    5    3    4    5    4    5     5
```

Probability calculations and combinatorics

- Other convenient functions to use include factorial and prod

```
factorial(5)
```

```
## [1] 120
```

```
prod(5:1)
```

```
## [1] 120
```

Discrete and continuous distributions

- R implements random sampling for most of the known standard discrete and continuous distributions:
- Discrete: Binomial distribution, geometric distribution, Poisson distribution
- Continuous: Normal, Beta, Gamma, log-normal, etc

Discrete and continuous distributions

Distribution	**R name**	**additional arguments**
beta	`beta`	shape1, shape2, ncp
binomial	`binom`	size, prob
Cauchy	`cauchy`	location, scale
chi-squared	`chisq`	df, ncp
exponential	`exp`	rate
F	`f`	df1, df2, ncp
gamma	`gamma`	shape, scale
geometric	`geom`	prob
log-normal	`lnorm`	meanlog, sdlog
logistic	`logis`	location, scale
negative binomial	`nbinom`	size, prob
normal	`norm`	mean, sd
Poisson	`pois`	lambda
Student's t	`t`	df, ncp
uniform	`unif`	min, max
Weibull	`weibull`	shape, scale

Discrete and continuous distributions

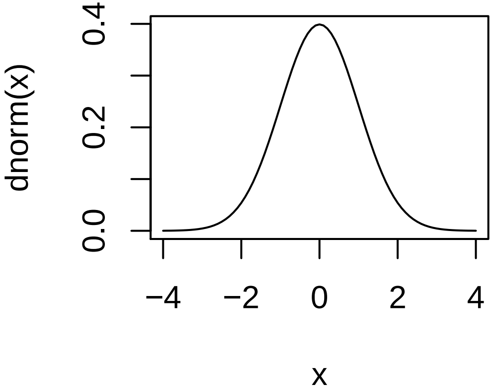
- Prefix the name given here by d for the density, `p` for the CDF, `q` for the quantile function and `r` for simulation (random deviates). The first argument is `x` for `dxxx`, `q` for `pxxx`, `p` for `qxxx` and `n` for `rxxx`. We next discuss and give some examples on these functions.

```
?rnorm
x=0
q=2
p=0.95
n=10
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)
```

Discrete and continuous distributions

- Densities

```r
x <- seq(-4,4,0.1)
plot(x,dnorm(x),type="l")
```

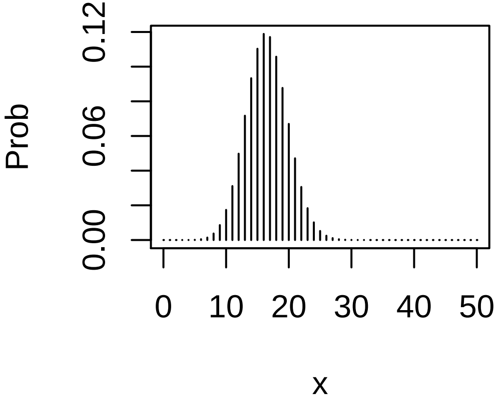



```r
curve(dnorm(x), from=-4, to=4)
```

Discrete and continuous distributions

- For discrete distributions, where variables can take on only distinct values, it is preferable to draw a pin diagram, here for the binomial distribution with n = 50 and p = 0.33:

```
x <- 0:50
plot(x,dbinom(x,size=50,prob=.33),type="h",ylab="Prob")
```



Discrete and continuous distributions

- Cumulative distribution functions

```
pnorm(160,mean=132,sd=13)
```

```
## [1] 0.9843739
```

```
pbinom(16,size=20,prob=.5)
```

```
## [1] 0.9987116
```

Discrete and continuous distributions

- Quantiles: If we have n normally distributed observations with the same mean μ and standard deviation σ, then it is known that the average $\bar{X}$ is normally distributed around μ with standard deviation $\sigma/\sqrt{n}$.

Discrete and continuous distributions

- A 95% confidence interval for μ can be obtained as:

$$\bar{X} - \sigma/\sqrt{n} \times z_{0.025} \leq \mu \leq \bar{X} + \sigma/\sqrt{n} \times z_{0.025}$$

where $z_{0.025}$ is the 2.5% upper quantile in the normal distribution.

```r
qnorm(0.025,lower.tail=FALSE)
```

```
## [1] 1.959964
```

```r
qnorm(0.975)
```

```
## [1] 1.959964
```

Discrete and continuous distributions

- Random numbers: Computer generates sequences of "pseudo-random" numbers, which for practical purposes behave as if they were drawn randomly.

```
rnorm(10,mean=7,sd=5)
rbinom(10,size=20,prob=.5)
```

Discrete and continuous distributions

- To lock the random seed, use `set.seed()`

```
set.seed(123)
rnorm(10,mean=7,sd=5)
rbinom(10,size=20,prob=.5)
```

Discrete and continuous distributions

- Exercise: Write a function to sample from binomial distribution without using rbinom and other binom functions.

Summary statistics for a single group

```r
set.seed(123)
x <- rnorm(50)
mean(x)
```

```
## [1] 0.03440355
```

```r
sd(x)
```

```
## [1] 0.92587
```

```r
var(x)
```

```
## [1] 0.8572352
```

```r
median(x)
```

```
## [1] -0.07264039
```

Summary statistics for a single group

```r
set.seed(123)
x <- rnorm(50)
quantile(x)
```

```
##          0%         25%         50%         75%        100%
## -1.96661716 -0.55931702 -0.07264039  0.69817699  2.16895597
```

```r
pvec <- seq(0,1,0.1)
quantile(x,pvec)
```

```
##          0%         10%         20%         30%         40%
## -1.96661716 -1.12461142 -0.68842368 -0.46849617 -0.29942796
##         60%         70%         80%         90%        100%
##  0.23594940  0.51467063  0.82482227  1.22705511  2.16895597
```

Summary statistics for a single group

- Exercise: Can you illustrate with simulations that if $X_i \sim N(\mu, \sigma^2)$ then $\bar{X} \sim N(\mu, \sigma^2/n)$

Summary statistics for a single group

- str()
- Data url: "http://www.ams.sunysb.edu/~pfkuan/Teaching/AMS597/Data/d_logret_6stocks.txt"

```
logret <- read.table("http://www.ams.sunysb.edu/~pfkuan/Teachi
str(logret)
```

```
## 'data.frame':    64 obs. of  7 variables:
##  $ Date     : chr  "1-Aug-00" "1-Sep-00" "2-Oct-00" "1-Nov-
##  $ Pfizer   : num  -0.00144 0.01749 -0.01705 0.01201 0.016
##  $ Intel    : num  0.05 -0.2556 0.0345 -0.0726 -0.1025 ...
##  $ Citigroup: num  0.0443 -0.0335 -0.0116 -0.0227 0.0107 ..
##  $ AmerExp  : num  0.0174 0.0127 -0.0049 -0.0383 0 ...
##  $ Exxon    : num  0.010225 0.037989 0.000331 -0.00365 -0.0
##  $ GenMotor : num  0.0933 -0.0322 -0.0196 -0.0949 0.0125 ..
```

Summary statistics for a single group

```r
names(logret)
```

```
## [1] "Date"       "Pfizer"     "Intel"      "Citigroup" "AmerEx
## [7] "GenMotor"
```

```r
logret$Intel[1:10]
```

```
##  [1]  0.04998126 -0.25561927  0.03454674 -0.07255067 -0.102
##  [7] -0.11219423 -0.03570214  0.06999448 -0.05826061
```

```r
sum(!is.na(logret$Intel))
```

```
## [1] 64
```

Summary statistics for a single group

```
summary(logret$Intel)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.255619 -0.026718 -0.014359 -0.005986  0.045387  0.12658
```

```
summary(logret)
```

```
##      Date              Pfizer                Intel
##  Length:64          Min.   :-0.060303   Min.   :-0.255619
##  Class :character   1st Qu.:-0.017109   1st Qu.:-0.026718
##  Mode  :character   Median :-0.002300   Median :-0.014359
##                     Mean   :-0.004041   Mean   :-0.005986
##                     3rd Qu.: 0.014631   3rd Qu.: 0.045387
##                     Max.   : 0.041784   Max.   : 0.126581
##    Citigroup             AmerExp               Exxon
##  Min.   :-0.0627462   Min.   :-0.0980439   Min.   :-0.0538
##  1st Qu.:-0.0221293   1st Qu.:-0.0115490   1st Qu.:-0.0062
##  Median : 0.0031789   Median : 0.0058001   Median : 0.0010
```

Graphical display of distributions

- Histograms, empirical distributions, Q-Q plot, Boxplot

```
x <- rnorm(100)
hist(x)
```

```
# empirical distribution function
n <- length(x)
plot(sort(x),(1:n)/n,type="s",ylim=c(0,1))
```

```
qqnorm(x)
```

```
boxplot(logret$Intel)
```

Graphical display of distributions

- The Central Limit Theorem (CLT) says that if X_i's are iid with mean μ and finite variance σ^2 (i.e, they need not be normally distributed), then
$$\frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \xrightarrow{d} N(0, 1)$$
as $n \to \infty$.
- Exercise: Illustrate the CLT with simulations using random uniform U(0,1) variables

Tables

- Categorical data are usually described in the form of tables.
- A two-way table can be entered as a matrix object.
- E.g.: Caffeine consumption by marital status

```
caff.marital <- matrix(c(652, 1537, 598, 242, 36, 46,
    38, 21, 218, 327, 106, 67), nrow = 3, byrow = T)
colnames(caff.marital) <- c("0", "1-150", "151-300",
    ">300")
rownames(caff.marital) <- c("Married", "Prev.married",
    "Single")
caff.marital
```

```
##                0 1-150 151-300 >300
## Married      652  1537     598  242
## Prev.married  36    46      38   21
## Single       218   327     106   67
```

Tables

- Furthermore, you can name the row and column names as follows. This is particularly useful if you are generating many tables with similar classification criteria.

```
names(dimnames(caff.marital)) <- c("marital", "consumption")
caff.marital
```

```
##                consumption
## marital           0 1-150 151-300 >300
##   Married       652  1537     598  242
##   Prev.married   36    46      38   21
##   Single        218   327     106   67
```

Tables

- Like any matrix, a table can be transposed with the `t` function.

```
t(caff.marital)
```

```
##              marital
## consumption Married Prev.married Single
##     0           652           36    218
##     1-150      1537           46    327
##     151-300     598           38    106
##     >300        242           21     67
```

Tables

- Exercise: Construct the following table which summarize the number of people smoking and nonsmoking in a class

	Smoking	Nonsmoking
Male	23	45
Female	34	54

Tables

- Exercise: First write a function which generates a matrix containing random uppercase letters of size $n \times p$. Then, write a function which returns the most frequent character for each row of such matrix.