STAT 201A, Fall 2012
R Basics

Angie Zhu

August 29, 2012

The purpose of this set of notes is to cover the commands required for STAT 201A. This is NOT a complete introduction.

1 Miscellanea

The R page on course website is at `http://www.stat.berkeley.edu/users/ani/s201af12/rpage.html`.

If you are familiar with MATLAB and new to R, this *MATLAB/R Reference* will be very helpful: `http://cran.r-project.org/doc/contrib/Hiebeler-matlabR.pdf`.

When you are using R build-in GUI, in order to evaluate these commands, position the cursor at the line you want to evaluate and hit "Command + Enter" for Macs and "Ctrl + R" for PCs.

If you prefer integrated development environment (IDE), *RStudio* is free downloadable at `http://rstudio.org/`.

If you are into `emacs`, check out `Emacs Speaks Statistics`.

2 Basics

We can use R as an oversized scientific calculator. The order of operations is important.

```
> 5 + 6 * 3

[1] 23

> exp(log(10))

[1] 10

> sin(2)^2 + cos(2)^2

[1] 1
```

Comments in R are started with "#." That is, lines starting with "#" will not be evaluated.

```
> # some comments
```

Let's try this out by creating a vector x

```
> x = 0:10
```

See what is stored in the vector x by typing x, and hitting "Enter"

```
> x

 [1]  0  1  2  3  4  5  6  7  8  9 10
```

There are other ways to create vectors:

```
> y = rep(2, 5)  # repeat 2 five times
> y

[1] 2 2 2 2 2

> z = seq(3, 6, by=0.25)
> z

 [1] 3.00 3.25 3.50 3.75 4.00 4.25 4.50 4.75 5.00 5.25 5.50 5.75 6.00

> w = c(23, 11, 29)  # concatenation
> w

[1] 23 11 29
```

x is now an "object" in R. "objects" are available to you in your current working environment. If you close the window without saving, it will be deleted and you will have no access to it the next time you open `R`

```
> objects()

[1] "w" "x" "y" "z"

> x

 [1]  0  1  2  3  4  5  6  7  8  9 10
```

The function `summary()` gives you a summary of the data

```
> summary(x)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0     2.5     5.0     5.0     7.5    10.0
```

Add 3 to each entry:

```
> x + 3

 [1]  3  4  5  6  7  8  9 10 11 12 13
```

Double each element of x:

```
> 2 * x

 [1]  0  2  4  6  8 10 12 14 16 18 20
```

Find the sum of all the numbers in x:

```
> sum(x)

[1] 55
```

Find out how many entries are in x:

```
> length(x)

[1] 11
```

Find the average of x:

```
> sum(x) / length(x)

[1] 5
```

Find the average of x using the `mean` function:

```
> mean(x)

[1] 5
```

Other useful functions are as follows:

```
> min(x)

[1] 0

> max(x)

[1] 10

> median(x)

[1] 5

> quantile(x)
```

```
  0%  25%  50%  75% 100%
 0.0  2.5  5.0  7.5 10.0
```

```
> range(x)
```

```
[1]  0 10
```

If you want to repeat the previous command, but just change a parameter, press the "up" arrow. R stores all your commands from the current session.

Anytime you want to know more about a particular command, you can type the command, preceded by a "?"

```
> ?mean
```

or

```
> help(mean)
```

If you are not sure of the command name, type

```
> help.start()
```

It will not work without the `()`. If you still cannot find out what you want, Google is your friend.

3 Naming Conventions

R is case sensitive. Object names (data or functions) should only contain alpha-numeric characters (A-Z, a-z, 0-9) or a period. Such names cannot start with a digit. You should avoid using object names that are already used by R, such as `t, c, q, T, F, ls, pt, mean, var, pi`, etc. Use descriptive names.

```
> pi
```

```
[1] 3.141593
```

Any object that you create using such system names would mask the built-in R object.

4 Probability Distributions

4.1 Binomial Distribution

Let $X \sim \text{Binomial}(n, p)$. $\Pr(X = x)$ for $x = 0, \ldots, n$ can be computed using the `dbinom()` function:

```
> dbinom(x=4, size=6, prob=0.5)
```

```
[1] 0.234375
```

where `size` and `prob` are the binomial parameters n and p, respectively.

The function `pbinom()` can be used to compute cumulative probabilities $\Pr(X \leq x)$:

```
> pbinom(4, 6, 0.5)

[1] 0.890625
```

Since `x`, `size`, and `prob` are the first three arguments of function `pbinom`, the names of the arguments can be omitted. However, the ordering of the pass-in arguments must respect the function definition.

To obtain quantiles, use function `qbinom()`:

```
> qbinom(0.89, 6, 0.5)

[1] 4
```

To generate binomial pseudorandom numbers, use function `rbinom()`:

```
> rbinom(n=3, size=6, prob=0.5)

[1] 2 3 2
```

```
> p = dbinom(0:6, 6, 0.5)
> barplot(p, space=0, names.arg=0:6, ylab="Probability",
+         main="Binomial(6, 0.5)", col="white")
```

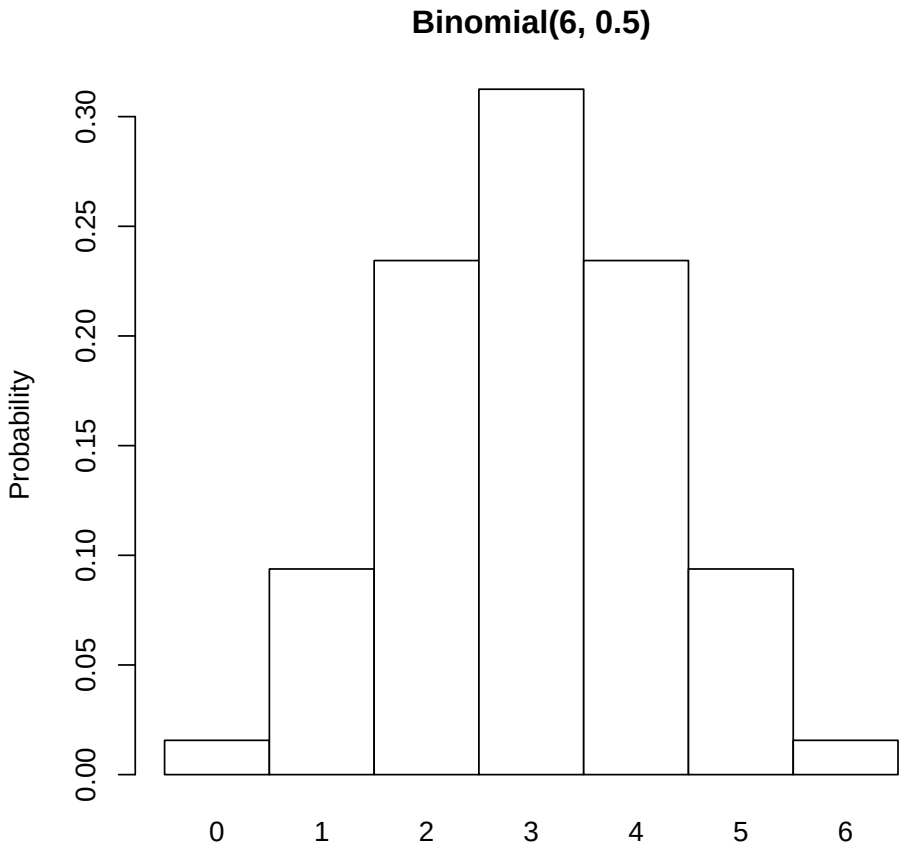

Figure 1: Binomial(6, 0.5) distribution.

4.2 Hypergeometric Distribution

$$\Pr(X = x) = \frac{\binom{m}{x}\binom{n}{k-x}}{\binom{m+n}{k}}$$

```
> dhyper(3, m=10, n=20, k=5)

[1] 0.1599933

> choose(10, 3) * choose(20, 2) / choose(30, 5)

[1] 0.1599933
```

Note that the function prefixes `d, p, q`, and `r` behave the same for other distributions, such as `dhyper, phyper, qhyper`, and `rhyper`.

```
> phyper(3, m=10, n=20, k=5)

[1] 0.9687592

> qhyper(0.96, m=10, n=20, k=5)

[1] 3

> rhyper(5, m=10, n=20, k=5)

[1] 1 1 2 2 2
```

```
> p = dhyper(0:5, m=10, n=20, k=5)
> barplot(p, space=0, names.arg=0:5, ylab="Probability",
+         main="Hypergeometric Distribution", col="white")
```

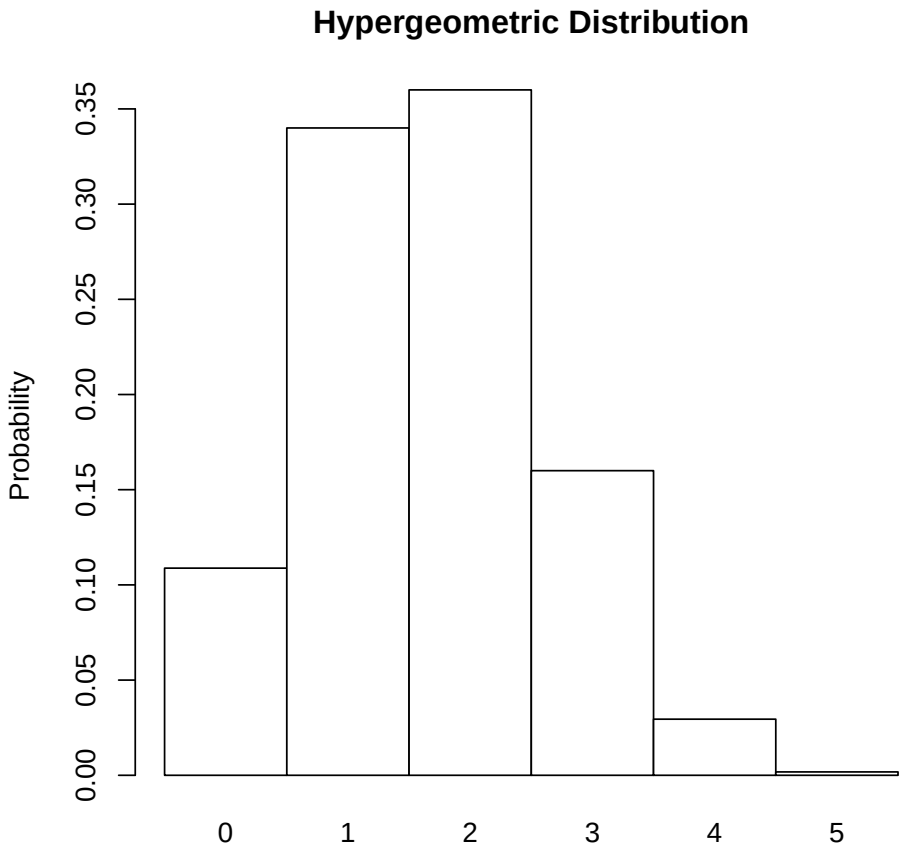

Figure 2: Hypergeometric distribution.

4.3 Poisson Distribution

Let $X \sim \text{Poisson}(\lambda)$. Then

$$\Pr(X = x) = \frac{e^{-x}\lambda^x}{x!}$$

```
> dpois(x=3, lambda=5)

[1] 0.1403739

> ppois(3, 5)
```

```
[1] 0.2650259
```

```
> qpois(0.26, 5)
```

```
[1] 3
```

```
> rpois(6, 5)
```

```
[1] 4 4 4 4 3 6
```

4.4 Normal Distribution

Let $X \sim \text{Normal}(\mu, \sigma^2)$. Then

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \ -\infty < x < \infty$$

```
> dnorm(0, mean=0, sd=5)
```

```
[1] 0.07978846
```

```
> pnorm(0, mean=0, sd=5)
```

```
[1] 0.5
```

```
> qnorm(0.975, 0, 1)
```

```
[1] 1.959964
```

```
> rnorm(10)  # sample from Standard Normal
```

```
 [1]  0.332471774 -0.671104707 -0.102770110  0.196642778 -1.273036572
 [6]  1.850053529  1.105719761 -1.448089764 -1.571379561 -0.002264943
```

Other continuous distribution will be covered later.

5 Simulation

For instance, let's get a random sample from Binomial(100, 0.5) of size 1000 and examine its histogram. Since binomial distributions can only have integer values, we need to modify the histogram to reflect this fact. Then superimpose its normal approximation Normal(50, 5).

```
> x = rbinom(1000, 100, 0.5)
> hist(x, freq=FALSE, breaks=seq(min(x)-0.5, max(x)+0.5, by=1),
+         main="Histogram of Random Sample from Binomial(100, 0.5)",
+         sub="Size 1000")
> w = seq(min(x), max(x), length.out=101)
> lines(w, dnorm(w, 50, 5), col="purple", lwd=2, lty=2)
> legend("topright", "Normal(50, 5)", col="purple", lty=2, lwd=2)
```

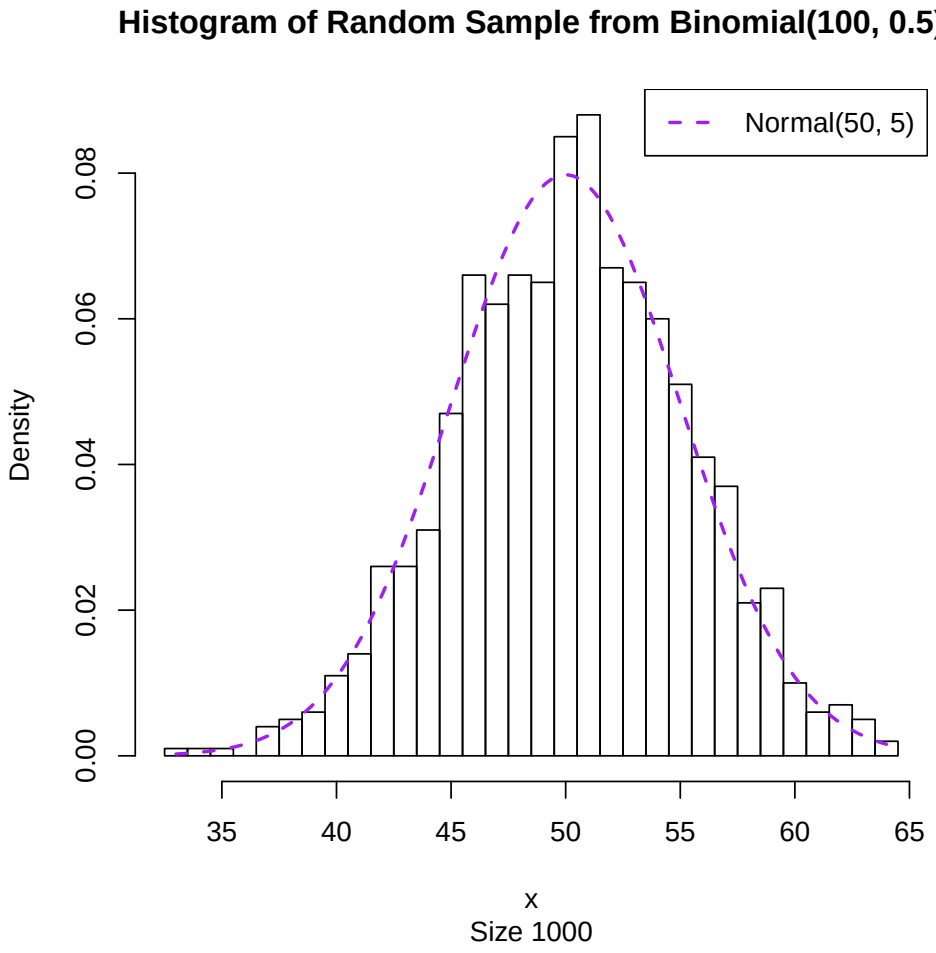

Figure 3: Histogram of Random Sample from Binomial(100, 0.5) of size 1000.

Let's look at a random sample from normal distribution:

```
> x = rnorm(1000, 3, 2)
> hist(x, freq=FALSE,
+         main="Histogram of Random Sample from Normal(3, 2) of size 1000")
> w = seq(min(x), max(x), length.out=101)
> lines(w, dnorm(w, 3, 2), col="purple", lwd=2, lty=2)
> legend("topright", "Normal(3, 2)", col="purple", lty=2, lwd=2)
```

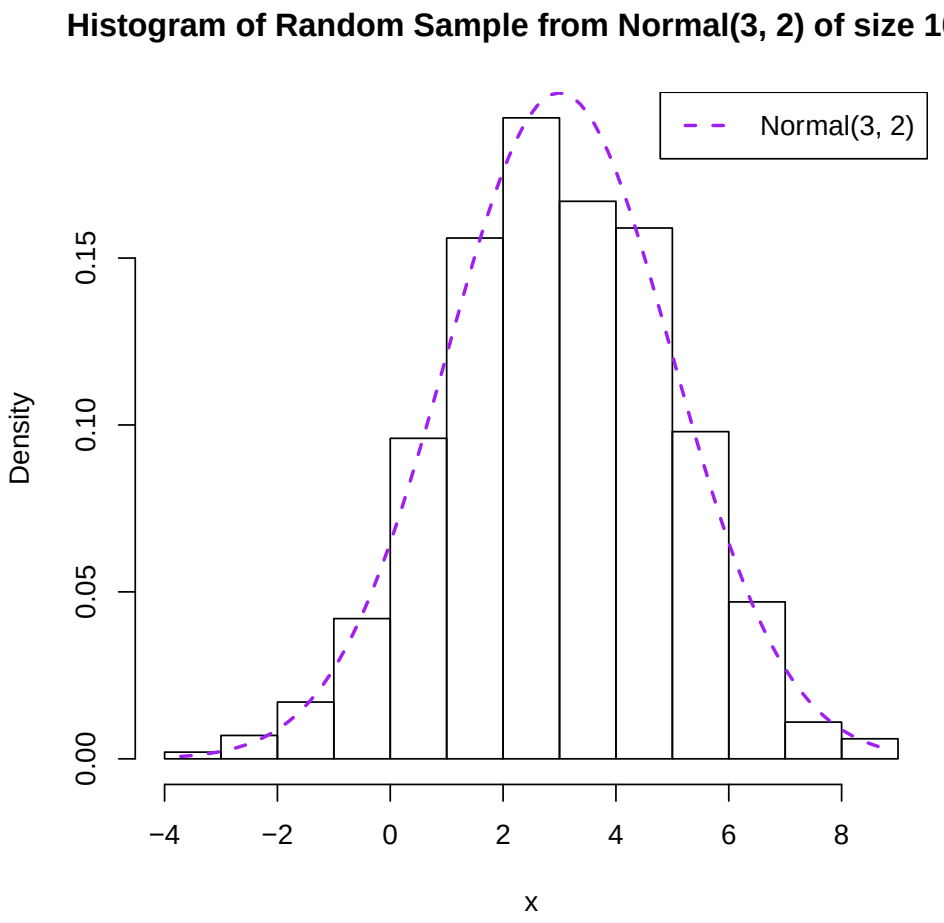

Figure 4: Histogram of Random Sample from Binomial(100, 0.5) of size 1000.

Other examples:

- To roll a fair die 3000 times:

  ```
  sample(6, size=3000, replace=TRUE)
  ```

- To choose 27 random numbers from 30 to 70:

```
sample(30:70, size=27, replace=TRUE)
```

- To flip a fair coin 1000 times:

```
sample(c("H", "T"), size=1000, replace=TRUE)
```