Handout #2

Lesson 1 (cont.): Working with numbers and generating data.

Distributions

For all of the standard statistical distributions, R includes functions to compute probabilities, densities, and quantiles. It can also generate random samples. Refer to p. 108 of Venables and Ripley for a table of distributions and the parameters they take.

Density values: *f(x)* for any *x* in the range of the random variable. Syntax: put a "d" in front of the abbreviation for the distribution. Example: dnorm(0,0,1) = 0.3989423 = 1/sqrt(2*pi)

Cumulative probability values: $F(x) = P(X \le x)$. Syntax: put a "p" in front of the abbreviation for the distribution. Example: pbinom(10,21,0.5) = 0.5.

Quantiles: $Q(u) = F^{-1}(u)$. Syntax: put a "q" in front of the abbreviation for the distribution. Example: qnorm(0.975,0,1) = 1.96.

Random data: Generate a sample of size *n* from a specified distribution.
Example: rpois(5,1) = 1 1 1 2 0.

Exercise 1: Generate a random sample of size 100 from the Normal(0,1) distribution. Find the probability of observing a value outside of the interval (-1.5, 1.5). What is the proportion of values outside of this interval in your sample?

Lesson 2: Looking at data.

Non-graphical commands

summary(x)
quantile(x,p)
mean(x)
var(x)
sd(x)
median(x)
stem(x)

Graphics

So far, we've only used the command line. But we can often learn a lot more by looking at our data graphically, and R gives us lots of ways to do this.

Getting started: When you use a command that generates a graphic, a new window will automatically open. Alternatively, you can type "windows( )" or "x11( )" to open a blank graphics window.

plot(x,y): The simplest plotting command.

Exercise 2: Generate a random sample of size 50 from the Uniform(0,5) distribution, call it "x." Use plot(x) to look at it. What happens? Now plot it as a sorted vector.

Options for plots. There are too many to deal with in one day, but we can get started with some of the main ones:

"type": The type of plot. Options include "p", "l", "b", "c", "o", "h", "s", and "n"
"col": The color of the plot symbols. Takes numeric values or words like "green"
"main": an overall title for the plot (in quotations)
"sub": a sub title for the plot (in quotations)
"xlab": a title for the x axis (in quotations)
"ylab": a title for the y axis (in quotations)
"cex": the size of the symbols in the plot

Example: plot(x, type= "b", col= "red", xlab= " ", ylab = "Data", main = "My Sample")

Exercise 3: Plot the Binomial(20, 0.25) density function (use type= "h" to make it look like a histogram). Label the "x" and "y" axes appropriately, and give it a title.

Boxplots.

boxplot(x) generates a graphical summary of a vector. There are many options.

Example:
x=rnorm(100,0,2)
boxplot(x)
boxplot(x, range=2)
boxplot(x, range=1)
boxplot(x, horizontal=T)

Exercise 4: Generate a random sample of size 100 from the Exponential(1) distribution. Look at the boxplot. What do you notice about it?

Quantile-quantile plots.

qqplot(x,y): Generate a quantile-quantile plot for two samples.
qqnorm(x): Compare the quantiles of a sample to quantiles from the normal distribution.
qqline(x): Add a line to the qqnorm plot.
q[dist](ppoints(x), ….): Generate matching quantiles for another target distribution.

Example:
x=runif(50,0,5).
qqnorm(x)
qqline(x)
qqnorm(x, datax=T)
qqline(x,datax=T)
y=qunif(ppoints(x))
qqplot(x,y,xlab= “Sample”, ylab=”Uniform Quantiles”)
lines(c(0,5),c(0,1))

Exercise 5: Generate a random sample of size 100 from the Exponential(1) distribution. Generate a qqnorm( ) plot and use qqline( ). What does this tell you about the differences between the two distributions? (*You can also type* plot(dexp(seq(0,5,0.05)),type="h") *to get an idea of the shape of the Exp(1) distribution.*)

Histograms.

hist(x) produces a histogram for a data vector. Lots of options.

Bin width: By default, R bases its bin width on Sturges’ formula:
$nclass = \lceil \log_2 n + 1 \rceil$
But this is often “adjusted” to make the bins look pretty.

Example:
x=runif(50)
log2(50)+1 = 6.643856 (*we should expect 7 bins*)
hist(x)
hist(x, nclass=7) (*try to force it!)*
hist(x,breaks=seq(0,1,length=8)) (*really force it!*)

Other options: Let *h* be the bin width.

$h = 3.5\hat{\sigma} n^{-1/3}$ (use nclass = “scott”, where $\hat{\sigma}$ is the sample standard deviation)
$h = 2Rn^{-1/3}$ (use nclass= “fd”, where *R* is the inter-quartile range)

In each case, the number of bins will be given by range(*x*)/*h*.

Example:
1/(3.5*sd(x)*50^{-1/3}) (*Compute the expected number of classes for the “scott” option*)
hist(x, nclass= “scott”)
1/(2*(quantile(x,.75,names=F)-quantile(x,.25,names=F))*50^{-1/3}) (*Compute the expected number of classes for the “fd” option*)
hist(x, nclass= “fd”)

Alternative: Venables and Ripley "truehist" function:

Example:
library(MASS)
x = rnorm(100)
truehist(x)
hist(x, nclass= "fd")
truehist(x, nbins= "fd")
hval=2*(quantile(x,.75,names=F)-quantile(x,.25,names=F))*100^{-1/3} (*Compute the bin width for the "fd" option*)
truehist(x,h=hval)

Exercise 6: The dataset "precip" provides the average amount of precipitation (rainfall) in inches for each of 70 United States (and Puerto Rico) cities.

Step 1: Load the data. Type data(precip) to load it into your R workspace.
Step 2: Analyze it. Use the tools we've learned to study the data. What can you say about it? How would you describe the distribution?

Exercise 7: CLT simulation. As we all know, the Central Limit Theorem states that the sampling distribution of the mean of samples of size *n* from an underlying population distribution with mean μ and standard deviation σ converges to a normal distribution with mean μ and standard deviation $\sigma/\sqrt{n}$. Your task is to write a function to simulate the CLT for samples from the Binomial(n_b,p) distribution where n_b is the number of trials and p is the probability of success on each trial. Your function should do the following:

(1) Generate k samples of size n from the underlying distribution.
(2) Compute the mean for each of the k samples.
(3) Produce a relative frequency histogram for the k means.
(4) Add the normal density curve for the appropriate normal distribution.

The values of n_b, p, n, and k should be specified as arguments to your function, not hard-coded (although default values should be specified for each of these arguments). One function that you might find helpful here is "apply." This lets you perform a specified operation on either the rows (selecting by specifying "margin = 1") or columns ("margin = 2") of a matrix. For example, suppose I want to compute the column sums of a matrix M. I would type "apply(M, margin=2, sum)."

Too easy? Then extend your function to allow for different input distributions, such as the uniform or the exponential…