R Commands

The following is an exhaustive list of the R commands you are responsible to know for the quizzes, midterm and final exam.

Note that most commands have many optional arguments. For example, the command `hist(x)` makes a basic histogram for a data set `x`. There are other arguments you can add to create a title for the histogram, to label the axes, to change the number of bars, to change the colour of the histogram, etc. You will only be responsible for the most basic commands that are given below, and not any arguments that are not mentioned.

Unit 1

For a data set `x`:

1. `hist(x)` creates a histogram of the data set.

2. (a) `boxplot(x)` creates an outlier boxplot of the data set.

   (b) `boxplot(x, range = 0)` creates a quantile boxplot of the data set.

   (c) `boxplot(x1, x2, ..., xn)` creates side-by-side outlier boxplots for the data sets `x1`, `x2`, ..., `xn`.

3. `mean(x)` calculates the sample mean.

4. (a) `sd(x)` calculates the sample standard deviation.

   (b) `var(x)` calculates the sample variance.

5. (a) `median(x)` calculates the median.

   (b) `fivenum(x)` calculates the five-number summary.

   (c) `IQR(x)` calculates the interquartile range.

*Note that if you calculate the interquartile range from Q1 and Q3 given in `fivenum`, you will usually get a slightly different result than using the `IQR(x)` command.

Unit 2

1. For a uniformly distributed variable X defined on the interval (a, b):

   (a) `punif(x, a, b)` calculates $P(X <$ `x`$)$.

   (b) `punif(x, a, b, lower.tail = FALSE)` or `1 - punif(x, a, b)` calculates $P(X >$ `x`$)$.

   (c) `punif(x2, a, b) - punif(x1, a, b)` calculates $P($`x1`$ < X <$ `x2`$)$.

   (d) `qunif(p, a, b)` calculates the p^{th} percentile of X (i.e., the value `x` such that $P(X <$ `x`$) = p$).

2. For a normally distributed variable X with mean μ and standard deviation σ:

   (a) `pnorm(x, μ, σ)` calculates $P(X <$ `x`$)$.

   (b) `pnorm(x, μ, σ, lower.tail = FALSE)` or `1 - pnorm(x, μ, σ)` calculates $P(X >$ `x`$)$.

   (c) `pnorm(x2, μ, σ) - pnorm(x1, μ, σ)` calculates $P($`x1`$ < X <$`x2`$)$.

   (d) `qnorm(p, μ, σ)` calculates the p^{th} percentile of X (i.e., the value `x` such that $P(X <$ `x`$) = p$).

   *Note that if no values are entered for μ or σ, R assumes a default of $\mu = 0$ and $\sigma = 1$, i.e., a standard normal variable Z.

3. `qqnorm(x)` creates a normal quantile plot for the data set `x`.

Unit 3

1. Let $\bar{X}$ based be the sample mean calculated from a sample of size n. If X follows a normal distribution with mean μ and standard deviation σ (or if the population is not normally distributed but $n \geq 30$):

   (a) `pnorm(xbar, μ, σ/sqrt(n))` calculates $P(\bar{X} < \texttt{xbar})$.

   (b) `pnorm(xbar, μ, σ/sqrt(n), lower.tail = FALSE)` or `1 - pnorm(xbar, μ, σ/sqrt(n))` calculates $P(\bar{X} > \texttt{xbar})$.

   (c) `pnorm(xbar2, μ, σ/sqrt(n)) - pnorm(xbar1, μ, σ/sqrt(n))` calculates $P(\texttt{xbar1} < \bar{X} < \texttt{xbar2}))$.

   (d) `qnorm(p, μ, σ/sqrt(n))` calculates the p^{th} percentile of $\bar{X}$ (i.e., the value `xbar` such that $P(X < \texttt{xbar}) = p$).

2. We take a (sufficiently large) sample of size n from some population with a proportion `p` of individuals that possess some characteristic and we calculate the sample proportion $\hat{p}$. Then:

   (a) `pnorm(phat, p, sqrt(p(1-p)/n))` calculates $P(\hat{p} < \texttt{phat})$.

   (b) `pnorm(phat, p, sqrt(p(1-p)/n), lower.tail=FALSE)` or `1 - pnorm(phat, p, sqrt(p(1-p)/n))` calculates $P(\hat{p} > \texttt{phat})$.

   (c) `pnorm(phat2, p, sqrt(p(1-p)/n)) - pnorm(phat1, p, sqrt(p(1-p)/n))` calculates $P(\texttt{phat1} < \hat{p} < \texttt{phat2}))$.

Units 4 & 5

If a variable X follows a normal distribution with **known** standard deviation σ, then based on a data set `x`:

1. `nsize(m, sigma = σ, conf.level = c)` calculates the sample size required to estimate μ to within a margin of error of `m` with confidence level `c`.

   *Note that if we do not specify a confidence level, R uses 95% as a default.

2. `z.test(x, sigma.x = σ, conf.level = c)` calculates a level `c` confidence interval for the population mean μ.

   *Note that if you do not include a the `conf.level` argument, R will calculate a 95% confidence interval by default.

   **Using this command will also produce output for a (nonsensical) hypothesis test – if you are only interested in a confidence interval, you can ignore this extra output.

3. `z.test(x, mu = μ0, sigma.x = σ, alternative = type)` will give the value of the test statistic and the P-value for a hypothesis test of H_0: $\mu = \mu 0$, where `μ0` is the value of μ under the null hypothesis.

   *Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `"less"` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

   **We can produce a confidence interval and conduct a hypothesis test using the following single command:

   ```
   z.test(x, mu = μ0, sigma.x = σ, alternative = type, conf.level = c)
   ```

Unit 6

1. For a t distribution with k degrees of freedom:

   (a) `pt(t, k)` calculates $P(T(k) < \texttt{t})$.

   (b) `pt(t, k, lower.tail = FALSE)` or `1 - pt(t, k)` calculates $P(T(k) > \texttt{t})$.

   (c) `pt(t2, k) - pt(t1, k)` calculates $P(\texttt{t1} < T(k) < \texttt{t2})$.

   (d) `qt(p, k)` calculates the p^{th} percentile of T (i.e., the value `t` such that $P(T(k) < \texttt{t}) = p$).

   If a variable X follows a normal distribution with **unknown** standard deviation, then based on a data set `x`:

2. `t.test(x, conf.level = c)` calculates a level `c` confidence interval for the population mean μ.

   *Note that if you do not include a the `conf.level` argument, R will calculate a 95% confidence interval by default.

   **Using this command will also produce output for a (nonsensical) hypothesis test – if you are only interested in a confidence interval, you can ignore this extra output.

3. `t.test(x, mu = μ0, alternative = type)` will give the value of the test statistic and the P-value for a hypothesis test of H_0: $\mu = \mu 0$, where `μ0` is the value of μ under the null hypothesis.

   *Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `"less"` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

   **We can produce a confidence interval and conduct a hypothesis test using the following single command:

   ```
   t.test(x, mu = μ0, alternative = type, conf.level = c)
   ```

Unit 7

1. `nsize(m, p = p*, conf.level = c, type = "pi")` calculates the sample size required to estimate a population proportion p to within a margin of error of `m` with confidence level `c`.

   *Note that if we do not specify a confidence level, R uses 95% as a default.

   **Note that if we do not specify a specific value of `p*`, R uses `p*` $= 0.5$ as a default.

2. We take a (sufficiently large) sample of size n from some population with **unknown** proportion `p` of individuals that possess some characteristic and we calculate the sample proportion $\hat{p}$. Then:

   `prop.test(x, n, p = p0, alternative = type, correct=FALSE)` will give the value of the test statistic and the P-value for a hypothesis test of H_0: $p =$ `p0`, where x is the number of successes, n is the number of trials, and p0 is the value of the population proportion under the null hypothesis. Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `"less"` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

*Note: The test statistic given by R is X^2. This is called a chi-square variable (which you will learn about in STAT 2150 if you take that course). The z statistic that we calculate is the square root of this X^2 value. (You don't need to know this on a quiz or exam if you are simply giving the R code to conduct the test.)

**R will also give a 95% confidence interval for p, but it uses a slightly different formula than we do to calculate the confidence interval. Therefore, the confidence interval given by R should be ignored.

Unit 8

1. `t.test(x, y, alternative = type, paired = TRUE)` will give the value of the test statistic and the P-value for a matched pairs t test of H_0: $\mu_d = 0$, where `x` and `y` are the names you've given to the two (dependent) data sets. Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `less` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

   *Note that using this command, R will also calculate a 95% confidence interval for μ_d. (If you are conducting a one-sided test, the interval should be ignored.) If you want a confidence interval with a different confidence level `c`, you can add the argument `conf.level = c` inside the parentheses.

2. `t.test(x, y, alternative = type, paired = FALSE, var.equal = TRUE)` will give the value of the test statistic and the P-value for a pooled two-sample t test of H_0: $\mu_X = \mu_Y$, where x and y are the names you've given to the data sets. Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `"less"` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

   *Note that using this command, R will also calculate a 95% confidence interval for $\mu_X - \mu_Y$. (If you are conducting a one-sided test, the interval should be ignored.) If you want a confidence interval with a different confidence level `c`, you can add the argument `conf.level = c` inside the parentheses.

3. `t.test(x, y, alternative = type, paired = FALSE, var.equal = FALSE)` will give the value of the test statistic and the P-value for an unpooled two-sample t test of H_0: $\mu_X = \mu_Y$, where x and y are the names you've given to the data sets. Note: In the place of `type`, enter `"greater"` to conduct an upper-tailed test, enter `"less"` to conduct a lower-tailed test, and enter `"two.sided"` to conduct a two-tailed test.

   *Note that using this command, R will also calculate a 95% confidence interval for $\mu_X - \mu_Y$. (If you are conducting a one-sided test, the interval should be ignored.) If you want a confidence interval with a different confidence level `c`, you can add the argument `conf.level = c` inside the parentheses.

   **Note: The degrees of freedom calculated by R for an unpooled test is different (a more precise estimate of the true degrees of freedom) than the estimate we use. Usually, the degrees of freedom in R will not be an integer.

Unit 9

We enter the sample data into R for an explanatory variable X and a response variable Y as vectors called `x` and `y`, respectively. Then:

1. `plot(x, y)` creates a scatterplot of `y` vs. `x`.

2. `cor(x, y)` calculates the correlation between `x` and `y`.

3. `lm(y ~ x)` calculates the equation of the least squares regression line.

4. `summary(lm(y ~ x))` calculates the value of the test statistic and the P-value for a test of H_0: $\beta_1 = 0$ vs. H_a: $\beta_1 \neq 0$, as well as the values of r^2 and the standard error s_e.