Analysis of Scientific Data

Summary of R functions

This sheet gives a summary of the functions in R that you will meet in the course this semester. You can also get more information using `help()` or `?` in R.

Importing Data

Use `data = read.csv("file")` for CSV data and `data = read.delim("file")` for tab-separated data.

Use `View(data)`, `str(data)`, `names(data)`, `nrow(data)` or `head(data)` as checks that your data was imported correctly.

Use `data$X`, for example, to extract the variable `X` from the data frame data.

Use `data$Z = data$Y1 – data$Y2`, for example, to calculate a new variable.

Use `subset(data, X == "x")` to extract a new data frame with just only the cases where `X` is `x`.

Use `data$X = ordered(data$X, c(A,B,C))` to make `X` an ordinal variable with `A` < `B` < `C`, for example.

Lattice Graphics

Use `library(lattice)` to load the lattice graphics library.

Strip plots	`stripplot(data$Y); stripplot(~ Y, data); stripplot(Y ~ X, data)`
Histogram	`histogram(data$Y); histogram(~ Y, data)`
Density plot	`densityplot(data$Y); densityplot(~ Y, data)`
Box plot	`bwplot(data$Y); bwplot(~ Y, data); bwplot(Y ~ X, data)`
Quantile plot	`qqmath(data$Y); qqmath(~ Y, data)`
Scatter plot	`xyplot(Y ~ X, data)`; use `type="p"` for data points, `"l"` for joining with lines, `"g"` for a grid, `"r"` for a regression line, and `"smooth"` for a smoothing line.
Bar chart	`barchart(table(data$X, data$Y))`
Spine plot	`spineplot(table(data$X, data$Y))`

Use the `group=Z` option to separate by variable `Z`. Add a title to your plot using the `main="title"` option. Change the axes labels using `xlab` and `ylab`. Get a simple legend with `auto.key=TRUE`.

Summary Statistics

Get basic statistics with `summary(data)`, `mean(data$Y)`, `median(data$Y)`, `sd(data$Y)`, `IQR(data$Y)`, and `fivenum(data$Y)`.

For categorical data use `table(data$Y)` or `table(data$Y, data$X)` for a two-way table. `prop.table()` can be applied to `table()` to get proportions and marginal proportions.

Use `aggregate()` to get statistics by group. For example, `aggregate(Y ~ X,data,mean)` gives the mean `Y` value for each category in `X`. This outputs a data frame which you can also use for plotting.

Basic Inference

One-sample t test	`t.test(data$X)`
Two-sample t test	`t.test(Y~X, data)` See `power.t.test()` for power calculations
One proportion	`prop.test(x, n)`
Two proportions	`prop.test(table(data$X, data$Y))`
Chi square test	`chisq.test(table(data$X, data$Y))`

Model Building

For each of the following you can use functions like `summary()`, `anova()`, `predict()`, and `residuals()` for details of the analysis.

Linear regression	`lm(Y ~ X, data)`
ANOVA	`aov(Y ~ X, data)`
Logistic regression	`glm(Y ~ X, data, family="binomial")`

Use `Y ~ X1*X2` for a two-factor model with an interaction term or `Y ~ X1+X2` for a model without an interaction term.

Other Calculations

Logarithms	`log(x)` for natural logs and `log10(x)` for base 10 logs
Exponentials	`exp(x)` for e^x and 10^x for 10^x
Factorial	`factorial(n)` gives $n!$
Combinations	`choose(n, k)`

Distributions

Note that the following distribution functions all give areas to the left (matching the theoretical definitions of these functions) whereas the tables in the textbook all give areas to the right (matching our use of the distributions).

Binomial	`dbinom(x,n,p)` for $P(X = x)$; `pbinom(x,n,p)` for $P(X \leq x)$
Normal	`pnorm(z)` for $P(Z \leq z)$; `qnorm(p)` for finding z such that $P(Z \leq z) = p$
T	`pt(t,df)` for $P(T_{\text{df}} \leq t)$; `qt(p,df)` for finding t such that $P(T_{\text{df}} \leq t) = p$
Chi square	`pchisq(x,df)` for $P(X_{\text{df}} \leq x)$; `qchisq(p,df)` for finding x such that $P(X_{\text{df}} \leq x) = p$
F	`pf(f,df1,df2)` for $P(F_{\text{df1,df2}} \leq f)$; `qf(p,df1,df2)` for finding f such that $P(F_{\text{df1,df2}} \leq f) = p$

Randomness

You can use `rbinom()`, `rnorm()`, etc. to generate sequences of random numbers from distributions.

Use `sample()` to generate a random sample of a certain size from a list of numbers or data.

Use `replicate()` to create a list by repeating a process multiple times.