Built-in Functions

Almost everything in R is done through functions. Here I'm only refering to numeric and character functions that are commonly used in creating or recoding variables.

(To practice working with functions, try the functions sections of this [this interactive course](https://www.datacamp.com/courses/intermediate-r-practice).)

Numeric Functions

|  |  |
| --- | --- |
| **Function** | **Description** |
| **abs(***x***)** | absolute value |
| **sqrt(***x***)** | square root |
| **ceiling(***x***)** | ceiling(3.475) is 4 |
| **floor(***x***)** | floor(3.475) is 3 |
| **trunc(***x***)** | trunc(5.99) is 5 |
| **round(***x***, digits=***n***)** | round(3.475, digits=2) is 3.48 |
| **signif(***x***, digits=***n***)** | signif(3.475, digits=2) is 3.5 |
| **cos(***x***), sin(***x***), tan(***x***)** | also acos(*x*), cosh(*x*), acosh(*x*), etc. |
| **log(***x***)** | natural logarithm |
| **log10(***x***)** | common logarithm |
| **exp(***x***)** | e^*x* |

Character Functions

|  |  |
| --- | --- |
| **Function** | **Description** |
| **substr(***x***, start=***n1***, stop=***n2***)** | Extract or replace substrings in a character vector. x <- "abcdef" substr(x, 2, 4) is "bcd" substr(x, 2, 4) <- "22222" is "a222ef" |
| **grep(***pattern***,***x***, ignore.case=**FALSE**, fixed=**FALSE**)** | Search for *pattern* in *x*. If fixed =FALSE then *pattern* is a [regular expression](http://regexlib.com/CheatSheet.aspx). If fixed=TRUE then *pattern* is a text string. Returns matching indices. grep("A", c("b","A","c"), fixed=TRUE) returns 2 |
| **sub(***pattern***,***replacement***,***x***, ignore.case =**FALSE**, fixed=**FALSE**)** | Find *pattern* in *x* and replace with *replacement* text. If fixed=FALSE then *pattern* is a regular expression[.](http://regexlib.com/CheatSheet.aspx) If fixed = T then *pattern* is a text string. sub("\\s",".","Hello There") returns "Hello.There" |
| **strsplit(***x***,***split***)** | Split the elements of character vector *x* at *split*. strsplit("abc", "") returns 3 element vector "a","b","c" |
| **paste(..., sep="")** | Concatenate strings after using *sep* string to seperate them. paste("x",1:3,sep="") returns c("x1","x2" "x3") paste("x",1:3,sep="M") returns c("xM1","xM2" "xM3") paste("Today is", date()) |
| **toupper(***x***)** | Uppercase |
| **tolower(***x***)** | Lowercase |

Statistical Probability Functions

The following table describes functions related to probaility distributions. For random number generators below, you can use set.seed(1234) or some other integer to create reproducible pseudo-random numbers.

|  |  |
| --- | --- |
| **Function** | **Description** |
| **dnorm(***x***)** | normal density function (by default m=0 sd=1) # plot standard normal curve x <- pretty(c(-3,3), 30) y <- dnorm(x) plot(x, y, type='l', xlab="Normal Deviate", ylab="Density", yaxs="i") |
| **pnorm(***q***)** | cumulative normal probability for q (area under the normal curve to the left of q) pnorm(1.96) is 0.975 |
| **qnorm(***p***)** | normal quantile. value at the p percentile of normal distribution qnorm(.9) is 1.28 # 90th percentile |
| **rnorm(***n***, m=**0**,sd=**1**)** | n random normal deviates with mean m and standard deviation sd. #50 random normal variates with mean=50, sd=10 x <- rnorm(50, m=50, sd=10) |
| **dbinom(***x***,***size***,***prob***) pbinom(***q***,***size***,***prob***) qbinom(***p***,***size***,***prob***) rbinom(***n***,***size***,***prob***)** | binomial distribution where size is the sample size and prob is the probability of a heads (pi) # prob of 0 to 5 heads of fair coin out of 10 flips dbinom(0:5, 10, .5) # prob of 5 or less heads of fair coin out of 10 flips pbinom(5, 10, .5) |
| **dpois(***x***,***lamda***) ppois(***q***,***lamda***) qpois(***p***,***lamda***) rpois(***n***,***lamda***)** | poisson distribution with m=std=lamda #probability of 0,1, or 2 events with lamda=4 dpois(0:2, 4) # probability of at least 3 events with lamda=4 1- ppois(2,4) |
| **dunif(***x***, min=**0**, max=**1**) punif(***q***, min=0, max=**1**) qunif(***p***, min=**0**, max=**1**) runif(***n***, min=**0**, max=**1**)** | uniform distribution, follows the same pattern as the normal distribution above. #10 uniform random variates x <- runif(10) |

Other Statistical Functions

Other useful statistical functions are provided in the following table. Each has the option na.rm to strip missing values before calculations. Otherwise the presence of missing values will lead to a missing result. Object can be a numeric vector or data frame.

|  |  |
| --- | --- |
| **Function** | **Description** |
| **mean(***x***, trim=**0**, na.rm=**FALSE**)** | mean of object x # trimmed mean, removing any missing values and # 5 percent of highest and lowest scores mx <- mean(x,trim=.05,na.rm=TRUE) |
| **sd(***x***)** | standard deviation of object(x). also look at var(x) for variance and mad(x) for median absolute deviation. |
| **median(***x***)** | median |
| **quantile(***x***,***probs***)** | quantiles where x is the numeric vector whose quantiles are desired and probs is a numeric vector with probabilities in [0,1]. # 30th and 84th percentiles of x y <- quantile(x, c(.3,.84)) |
| **range(***x***)** | range |
| **sum(***x***)** | sum |
| **diff(***x***, lag=***1***)** | lagged differences, with lag indicating which lag to use |
| **min(***x***)** | minimum |
| **max(***x***)** | maximum |
| **scale(***x***, center=**TRUE**, scale=**TRUE**)** | column center or standardize a matrix. |

Other Useful Functions

|  |  |
| --- | --- |
| **Function** | **Description** |
| **seq(***from***,***to***,***by***)** | generate a sequence indices <- seq(1,10,2) #indices is c(1, 3, 5, 7, 9) |
| **rep(***x***,***ntimes***)** | repeat *x* *n* times y <- rep(1:3, 2) # y is c(1, 2, 3, 1, 2, 3) |
| **cut(***x***,***n***)** | divide continuous variable in factor with *n* levels y <- cut(x, 5) |