Based closely from an excerpt of *R Visualization: Derive Meaning from Data*, David Gerbing, CRC Press, May, 2020, pp 199-200 (approximate)

8.2.4 Data Preparation for Date Variables

To read dates into R, consider two possibilities.

- Read the dates from **character** strings as stored in usually a plain text file, or sometimes an Excel data file.
- Read from an Excel file that has stored the dates as the Excel **Date** type.

These two possibilities are discussed next.

Convert Character Strings to Dates

For text data files, comma (csv) or tab delimited, all data values, including dates, are stored as character strings. That is the definition of a text file, a collection of plain, unformatted, individual characters.

When reading data from a text file, various R read functions automatically convert data values from character strings to their corresponding numeric type for variables exclusively with values that represent numbers. However, when R reads data values from a text file as character strings that represent dates, there is no automatic conversion to an R date format, such as type **Date**. Instead, by default R reads the dates into the data frame as type **character**. That is, R interprets the dates such as "7/1/2015" read from text data files no differently from any other character string, such as "ABC" or "Hi There".

The method discussed here transforms the data values of dates *after* they are read into R as character strings to the R variable type **Date**. Invoke the function **as.Date()** to convert the data values of a variable of type **character** to type **Date**. The complication is that many different expressions of a date exist, such as, for the first day of July, 2015, "July 1, 2015" or "7/1/2015" or "1-7-15", and many other possibilities.

The key to the date conversion is the **format** parameter, which links each part of the date character string – year, month and day – to the corresponding part of the date with a corresponding formatting code, presented in the order in which the parts of the date appear in the corresponding character string. R needs to be instructed as to how a particular date is represented in the text data file. The **format** parameter provides that instruction.

Each formatting code begins with the % sign. The code **%Y** identifies a four-digit year in the character string that contains a date. Identify a year expressed as two digits with **%y**. To identify a month expressed numerically, as digits from 1 to 12, use **%m**. Identify a month spelled out alphabetically in full with **%B**, and an abbreviated alphabetic month with **%b**. Identify the day of a month with **%d**. Include any delimiters in the character string such as a hyphen, forward slash, blank, or comma in the format expression in the same position found in the character string.

R has a default date format, the ISO 8601 international standard, defined by a four-digit year, a hyphen, a two-digit month, a hyphen, and then a two-digit day. For example, express the first day of July, 2015 with this standard as "2015-07-01". Dates expressed as character strings in this format, or the closely related form with forward slashes in place of the hyphens, need no explicit format parameter in the call to **as.Date()** to convert to type *Date*. All other date representations embedded in a character string require the format parameter for R to correctly transform a string of characters to an R date.

Listing 8.3 presents a variety of character string expressions of dates, all of which output to the same date, "2015-09-01" when the proper format is applied.

```
as.Date() function calls that transform character strings to the same date: "2015-09-01"
as.Date("2015-09-01")
as.Date("2015-09-01", format="%Y-%m-%d")
as.Date("2015/9/1")
as.Date("2015.09.01", format="%Y.%m.%d")
as.Date("09/01/2015", format="%m/%d/%Y")
as.Date("9/1/15", format="%m/%d/%y")
as.Date("September 1, 2015", format="%B %d, %Y")
as.Date("Sep 1, 2015", format="%b %d, %Y")
as.Date("20150901", format="%Y%m%d")
```

Listing 8.3: Alternate expressions of dates in character strings defined in the **strptime()** help file that decode to the "2015-09-01" with a call to as.Date() with the proper format.

To apply these calls to **as.Date()** to transform all the data values for a variable, first read the data values stored in the web text file PPStechLong.csv into the R data frame *d*.

```
d <- Read("http://lessRstats.com/data/PPStechLong.csv")
```

The output from **Read()** shows us the variables that it just read, as well as the data type for how the data values for these variables are represented in R. The output here indicates that the variable named *date* has been read as type **character**.

```
    Variable                   Missing  Unique
        Name      Type  Values  Values  Values   First and last values
-------------------------------------------------------------------------
 1      date character    1374       0     458   12/1/1980  1/1/1981 ...
 2   Company character    1374       0       3   Apple  Apple ... Intel
 3     Price    double    1374       0    1259   0.027  0.023 ... 46.634
-------------------------------------------------------------------------
```

Before we can covert the character string to type **Date** we need to know the format of the dates. **Read()** provides sample data values, so examine its output. The first date value for the

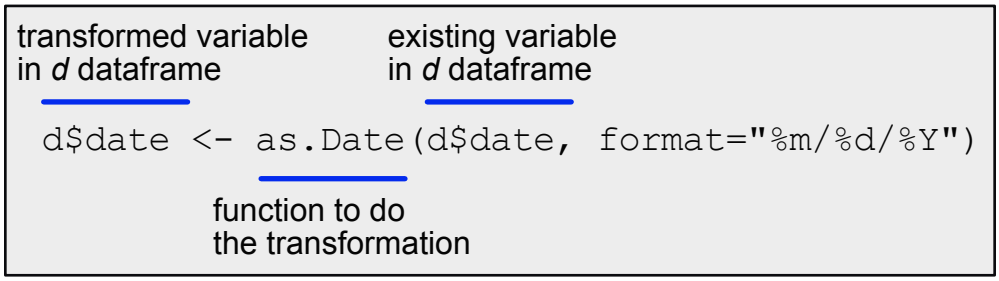
variable *date* is the character string 12/1/1980. We see that the date is formatted as number of the month, number of the day of the month, and then a four-digit year.

Referring back to Listing 8.3 above, we see the following line, which converts one specific date stored as a **character** string, "9/1/15", to an R **Date** data type.

```
as.Date("9/1/15", format="%m/%d/%y")
```

The R format of dates in this form is `"%m/%d/%y"`. Ultimately, convert all the data values for a variable, not just one specific date. To do so, apply the date format to the entire column of data values for the variable *date*.

Convert the variable named *date* in the *d* data frame from type **character** to the **Date** type as follows. Identify the variable by both its name and the data frame in which it is stored. The R notation to identify a variable is data frame name, $, variable name, in this example, *d$date*, which refers to the variable named *date* in the *d* data frame.



This example takes the date values for the variable *date* stored in the *d* data frame, applies the **as.Date()** transformation with the specified format, and then names the transformed variable with the same name, *d$date*.

To verify that the correct date format has been specified so that the date conversion was correctly accomplished, can use the **lessR** function **details()**, with abbreviation **db()** fto invoke the brief version. The output of **db()** provides the same information that **Read()** provides. The name of the data frame appears between the parentheses, with *d* as the default value. No need to specify the name unless named something other than *d*. Look for both the correct display of the date as well as the variable type of **Date**.

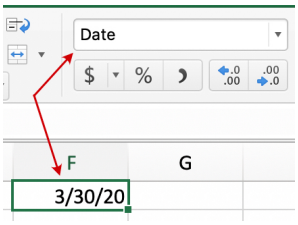
```
> db()
```

```
    Variable                   Missing  Unique
        Name      Type  Values  Values  Values   First and last values
-------------------------------------------------------------------------
 1      date      Date    1374       0     458   1980-12-01 ... 2019-01-01
 2   Company character   1374       0       3   Apple  Apple ... Intel
 3     Price    double   1374       0    1259   0.027  0.023 ... 46.634
-------------------------------------------------------------------------
```

After the **as.Date()** transformation, the type of the variable date is now **Date**. Further, the dates are now stored in the R default ISO standard of a four-digit year, two-digit month, and two-digit month, each separate by a hyphen.

Read Excel Date Type Directly as Dates

With Excel files, dates can be represented as an Excel **Date** type, which Excel does automatically when it recognizes a character string as a date. Enter the characters of a date into Excel such as from the keyboard or read from a text data file. As shown with the screen pic of an Excel worksheet, from the characters that form a date, Excel automatically converts the character string it recognizes as a date to a **Date** type.



When an Excel file with dates formatted as the **Date** type, R then properly interprets the dates as dates when read into R.

Internal Storage of Dates

Both R and Excel store dates internally as integers, called serial numbers, sequentially numbered by day from an arbitrarily set origin. R uses the origin "1970-01-01", which corresponds to serial number 1. Windows and Macintosh Excel versions originally managed to base their date systems on different origins. Recent versions of Macintosh Excel (since 2011) standardize on the Windows origin, which is "1900-01-01. Earlier versions of Macintosh Excel used "1904-01-01".

For reference, **as.Date()** also converts serial numbers to R **Date** fields. To convert, provide the serial number as an integer and then the corresponding origin. Listing 8.4 shows this conversion for the Excel origin and the R origin.

R Input Different integers (serial numbers) that each output "2015-09-01".
as.Date(42246, origin="1900-01-01")
as.Date(16679, origin="1970-01-01")

Listing 8.4: Alternate expressions of dates as integers that decode to the "2015-09-01" with a call to **as.Date()** with the proper origin.

The serial number 42246 corresponds to the date of Sept 1, 2015 in Excel. The serial number 16679 corresponds to the same date in R.