**Ghement Statistical Consulting Company Ltd. © 2013**

**Reading Time-Stamped Dates in R**

Assume you have a comma delimited data file (.csv) which includes a variable named **Date**, such that this variable represents a calendar date accompanied by a time stamp. Some possible values for the **Date** variable are listed below:

2009-12-25 18:39:11

2009-12-25 18:39:12

2009-12-25 18:39:13

Each of these time-stamped dates is of the form:

Year-Month-Day Hour-Minute-Second

How can you instruct R to read the data file and treat the **Date** variable as a time-stamped date? The answer is simple:

1. Read the csv data file into R using the read.csv() function with the option as.is=TRUE;
2. Check that the **Date** variable is treated as a character variable by R;
3. Use the function strptime() available in the **zoo** package to convert the Date variable into a time-stamped date.

**Example:**

To illustrate this answer, let’s try to read the data file **FileWithDates.csv** into R, which contains a variable named **Date** taking the values mentioned above.

dataset <- read.csv("FileWithDates.csv", as.is=TRUE)

dataset

> dataset

Date

R Output

1 25/12/2009 18:39:11

2 25/12/2009 18:39:12

3 25/12/2009 18:39:13

We now examine the structure of the R data frame named **dataset**, which stores the data in **FileWithDates.csv**.

str(dataset)

> str(dataset)

'data.frame': 3 obs. of 1 variable:

$ Date: chr "25/12/2009 18:39:11" "25/12/2009 18:39:12" "25/12/2009 18:39:13"

Clearly, R treats the **Date** variable as a character variable. However, we would like to treat this variable internally as time-stamped date variable.

To convert Date from a character variable to a time-stamped date variable, we use the strptime() function in the zoo package:

require(zoo)

dataset$Date <- strptime(dataset$Date, format = "%d/%m/%Y %H:%M:%S")

dataset

> dataset

Date

1 2009-12-25 18:39:11

2 2009-12-25 18:39:12

3 2009-12-25 18:39:13

If we check the structure **dataset** again we will see that, internally, R treats the **Date** variable as a time-stamped date variable (as revealed by the POSIXlt format of the **Date** variable):

dataset

> str(dataset)

'data.frame': 3 obs. of 1 variable:

$ Date: POSIXlt, format: "2009-12-25 18:39:11" "2009-12-25 18:39:12" "2009-12-25 18:39:13"

For more details on the POSIXlt format, which represents calendar dates and times (to the nearest second), you can use the R command:

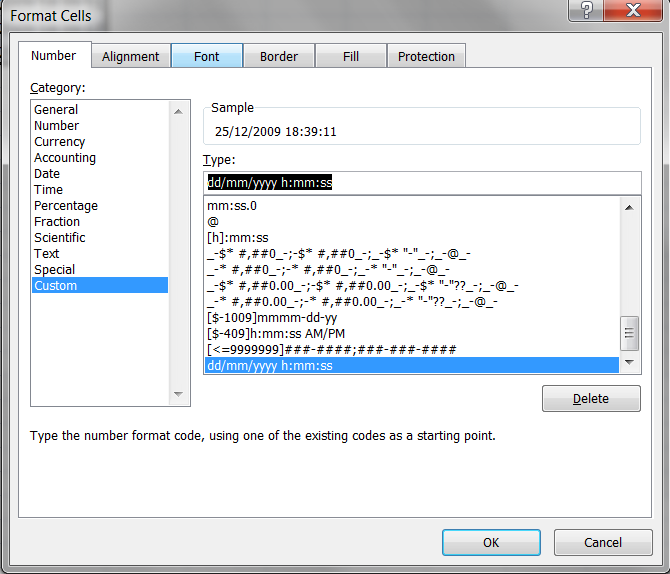
help(POSIXlt)

**Important Comments**

1. By default, Excel stores time-stamped data using the format **date** + **time**, where time allows representation of hours and minutes only. If your time stamp includes seconds, you need to change this default in order to ensure correct export of the data from Excel to R.

For **FileWithDates.csv**,you can change Excel’s default by selecting all cells of the **Date** variable,

right-clicking on these cells, choosing the **Custom** format and manually changing the **Type** of the format from dd/mm/yyyy h:mm to dd/mm/yyyy h:mm:ss (as seen in the screenshot below). Click **OK** to save this change and then proceed to save the data file as a csv file.



1. When R reads csv data files into R, it may change the format in which it stores time-stamped dates. For this reason, it is important to examine the structure of the imported data set in R to determine which format is used by R for the storage of dates and times. In the example given above, R stored dates and times as:

25/12/2009 18:39:11

This is why we called strptime() using the option format = "%d/%m/%Y %H:%M:%S".

1. The function strptime() simply strips date and time information from a character variable in order to produce a genuine time-stamped date in R.

"%d/%m/%Y %H:%M:%S".

From the help file for strptime(), we learn that:

%d Day of the month as decimal number (01–31).

%m Month as decimal number (01–12).

%Y Year with century.

%H Hours as decimal number (00–23). As a special exception times such as

24:00:00 are accepted for input, since ISO 8601 allows these.

%M Minute as decimal number (00–59).

%S Second as decimal number (00–61), allowing for up to two leap-seconds

(but POSIX-compliant implementations will ignore leap seconds).

Note that the help file for strptime() can be invoked using the R command:

help(strptime, package="zoo")

1. The format option of strptime() is determined by the specific nature of the time-stamped data.

In particular, the separators between %d, %m and %Y can be not only a forward slash, but also a white space, a comma or a dash:

"%d %m %Y %H:%M:%S" (white space separates day, month and year);

"%d,%m,%Y %H:%M:%S" (comma separates day, month and year);

"%d-%m-%Y %H:%M:%S" (dash separates day, month and year).

Similarly, the symbol which separates the hour, minutes and seconds can be a white space, a comma or a dash.