Professor Bear :: Excel files in R

Bear

September 1, 2018

It is easier and more common to work with comma-delimited text files (`.csv`), and tab-delimited text files than with native Excel files (`.xlsx`)nut there are options for bringing data in from `.xlsx` files, too.

Additional packages needed

To run the code in the lesson you may need additional packages.

- If necessary install the following packages.

```
install.packages("gdata");
```

Data

We'll be using GDP per capita, life expectancy, infant.mortality, and literacy data made availble by the WorldBank data.worldbank.org

GDP per capita (current US$)

GDP per capita is gross domestic product divided by midyear population. GDP is the sum of gross value added by all resident producers in the economy plus any product taxes and minus any subsidies not included in the value of the products. It is calculated without making deductions for depreciation of fabricated assets or for depletion and degradation of natural resources. Data are in current U.S. dollars.

Life expectancy at birth, total (years)

Life expectancy at birth indicates the number of years a newborn infant would live if prevailing patterns of mortality at the time of its birth were to stay the same throughout its life. Derived from male and female life expectancy at birth from sources such as: (1) United Nations Population Division. World Population Prospects, (2) United Nations Statistical Division. Population and Vital Statistics Report (various years), (3) Census reports and other statistical publications from national statistical offices, (4) Eurostat: Demographic Statistics, (5) Secretariat of the Pacific Community: Statistics and Demography Programme, and (6) U.S. Census Bureau: International Database.

Mortality rate, infant (per 1,000 live births)

Infant mortality rate is the number of infants dying before reaching one year of age, per 1,000 live births in a given year. Estimates developed by the UN Inter-agency Group for

Child Mortality Estimation (UNICEF, WHO, World Bank, UN DESA Population Division) at www.childmortality.org.

Literacy rate, adult total (% of people ages 15 and above)

Adult (15+) literacy rate (%). Total is the percentage of the population age 15 and above who can, with understanding, read and write a short, simple statement on their everyday life. Generally, 'literacy' also encompasses 'numeracy', the ability to make simple arithmetic calculations. This indicator is calculated by dividing the number of literates aged 15 years and over by the corresponding age group population and multiplying the result by 100.

```
suppressPackageStartupMessages(library(gdata))
```

(Clean) .xlsx

A "clean" `.xlsx` file is one where the data is in a fairly raw form. That is, it's not a workbook with multiple tables and tabs. Essentially a "clean" `.xlsx` file is one that could be saved to a single `.csv` with no data loss.

`gdata` package

gdata: Various R Programming Tools for Data Manipulation

Various R programming tools for data manipulation, including: - medical unit conversions ('ConvertMedUnits', 'MedUnits'), - combining objects ('bindData', 'cbindX', 'combine', 'interleave'), - character vector operations ('centerText', 'startsWith', 'trim'), - factor manipulation ('levels', 'reorder.factor', 'mapLevels'), - obtaining information about R objects ('object.size', 'elem', 'env', 'humanReadable', 'is.what', 'll', 'keep', 'ls.funs', 'Args','nPairs', 'nobs'), - manipulating MS-Excel formatted files ('read.xls', 'installXLSXsupport', 'sheetCount', 'xlsFormats'), - generating fixed-width format files ('write.fwf'), - extricating components of date & time objects ('getYear', 'getMonth', 'getDay', 'getHour', 'getMin', 'getSec'), - operations on columns of data frames ('matchcols', 'rename.vars'), - matrix operations ('unmatrix', 'upperTriangle', 'lowerTriangle'), - operations on vectors ('case', 'unknownToNA', 'duplicated2', 'trimSum'), - operations on data frames ('frameApply', 'wideByFactor'), - value of last evaluated expression ('ans'), and - wrapper for 'sample' that ensures consistent behavior for both scalar and vector arguments ('resample')

The documentation is at https://cran.r-project.org/web/packages/gdata/gdata.pdf

```
# Load our data
dwb <- read.xls("data.worldbank.org.ds.xlsx", sheet = 1, header = TRUE)
head(dwb)
```

```
##          Country Country.Code                     Region
## 1    Afghanistan          AFG                 South Asia
## 2        Albania          ALB      Europe & Central Asia
## 3        Algeria          DZA Middle East & North Africa
## 4 American Samoa          ASM        East Asia & Pacific
## 5        Andorra          ADO      Europe & Central Asia
## 6         Angola          AGO         Sub-Saharan Africa
##         Income.Group Per.capita.income Literacy Life.expectancy
## 1          Low income       590.2695154       ..     60.37446341
## 2 Upper middle income       3965.016806       ..     77.83046341
## 3 Upper middle income       4206.031232       ..     74.80809756
## 4 Upper middle income                ..       ..              ..
## 5         High income                ..       ..              ..
## 6 Upper middle income        4102.11859 70.77841     52.26687805
##   Infant.mortality
## 1             66.3
## 2             12.5
## 3             21.9
## 4               ..
## 5              2.1
## 6               96
```

Note the `..` in the empty values and how gdata can clean it up for analyis in R while importing. That is many R functions can handle NAs but would choke on `..` representing an empty value.

```
dwb <- read.xls("data.worldbank.org.ds.xlsx", sheet = 1, header =
TRUE,na.strings=c("NA","..", "?"))
head(dwb)
```

```
##          Country Country.Code                     Region
## 1    Afghanistan          AFG                 South Asia
## 2        Albania          ALB      Europe & Central Asia
## 3        Algeria          DZA Middle East & North Africa
## 4 American Samoa          ASM        East Asia & Pacific
## 5        Andorra          ADO      Europe & Central Asia
## 6         Angola          AGO         Sub-Saharan Africa
##         Income.Group Per.capita.income Literacy Life.expectancy
## 1          Low income          590.2695       NA        60.37446
## 2 Upper middle income         3965.0168       NA        77.83046
## 3 Upper middle income         4206.0312       NA        74.80810
## 4 Upper middle income                NA       NA              NA
## 5         High income                NA       NA              NA
## 6 Upper middle income         4102.1186 70.77841        52.26688
##   Infant.mortality
## 1             66.3
## 2             12.5
## 3             21.9
## 4               NA
## 5              2.1
## 6             96.0
```

Copy and paste from Excel to R

You can copy and paste from Excel to R but ut has drawbacks: it is hard to automate, and it requires an open Excel file to select data and copy.

```r
df = read.table("clipboard")
```

Exporting to Excel

Exporting data to Excel can be done as a *.csv* or as a *.xlsx*

.csv

Simple one sheet data frames can be exported as a simple .csv file.

- `write.csv()` – simply specify what to output and the filename to which to output it. Type `?write.csv` for the complete documentation
- `write.csv2()` – just like `read.csv2()`, `write.csv2()` is designed for use in countries where a comma is used for a decimal point and a semicolon is used as the delimiter.

.xlsx

If you have multiple data frames that you want to place on separate tabs in a single workbook, the `WriteXLS` package (`install.packages("WriteXLS")` and then `library(WriteXLS)` is used for this.

```r
# Create a vector with the names of the data frame objects
sheet.data <- c("df1", "df2", "df3")

# Create a vector with the worksheet names we want to use
sheet_names <- c("Data Frame 1", "Data Frame 2", "Data Frame 3")

# Write out an Excel file
WriteXLS(sheet_data,
         ExcelFileName = "output.data.xlsx",
         SheetNames = sheet_names)
```