

Introduction to Importing and Managing Financial Data

Presented at Saint Louis RUG
August 17, 2016

All data are wrong...



...but some are useful

- Data munging is 80% of most analysis
- Raw time series data come in various formats, shapes, sizes, and periodicities
- Combining data from several different sources can be difficult

Quandl and quantmod

- Quandl
 - One central database
 - One main function: `Quandl::Quandl`
- quantmod
 - No database (all data are from external providers)
 - Main function: `quantmod::getSymbols`
 - “Dispatches” to “methods” for specific data providers

quantmod::getSymbols

- Consistent interface to various data sources
 - `Symbols` argument identifies the instruments to load
 - `src` argument specifies the data source
- Behaves like `base::load`
 - Automatically creates objects in an environment
 - Set `auto.assign = FALSE` to return the data instead
- Creates xts objects by default

quantmod::getSymbols

- Consistent interface to various data sources
 - `Symbols` argument identifies the instruments to load
 - `src` argument specifies the data source
 - Sometimes data may not be available
- Behaves like `base::load`
 - Automatically creates objects in an environment
 - Set `auto.assign = FALSE` to return the data instead
- Creates xts objects by default

Quandl::Quandl

- One function for all databases
 - `code` argument specifies both source and instruments
 - `code` = “database/dataset”
- Behaves like a “normal” function and returns data
- Returns data.frame objects by default

quantmod column extractors

- Extract one column: Op, Hi, Lo, Cl, Vo, Ad

```
> getSymbols("SPY")  
[1] "SPY"  
> head(Cl(SPY))  
                SPY.Close  
2007-01-03      141.37  
2007-01-04      141.67  
2007-01-05      140.54  
2007-01-08      141.19  
2007-01-09      141.07  
2007-01-10      141.54
```


quantmod column extractors

- Extract several columns: OHLC, HLC, OHLCV

```
> getSymbols("SPY")
```

```
[1] "SPY"
```

```
> head(HLC(SPY))
```

	SPY.High	SPY.Low	SPY.Close
2007-01-03	142.86	140.57	141.37
2007-01-04	142.05	140.61	141.67
2007-01-05	141.40	140.38	140.54
2007-01-08	141.41	140.25	141.19
2007-01-09	141.60	140.40	141.07
2007-01-10	141.57	140.30	141.54

quantmod column extractors

- Extract specific column: getPrice

```
> # Download CME data for CL and BZ
> codes <- c("CME/CLH2016", "CME/BZH2016")
> oil_data <- Quandl(code = codes, type = "xts")
> # Extract the Open price for CLH2016
cl_open <- getPrice(oil_data,
                    symbol = "CLH2016",
                    prefer = "Open")
```

Quandl Transformations

- Quandl provides built-in:
 - Transformations
 - transform argument
 - "diff", "rdiff", "normalize", "cumul", "rdiff_from"
 - Aggregations
 - collapse argument
 - "daily", "weekly", "monthly", "quarterly", "annual"

quantmod transformations

- quantmod relies on xts for transformations
 - `xts::to.period`
 - `xts::period.apply`
 - `xts::lag`

Setting getSymbols defaults

- Customize defaults for getSymbols...

> # Pull from Google Finance by default
> setDefaults(getSymbols, src = "google")
> # Get GOOG data
> getSymbols("GOOG")
[1] "GOOG"
> # Verify data was pulled from Google
> attr(GOOG, "src")
[1] "google"

Setting getSymbols defaults

- ...or any getSymbols “method”

```
> setDefaults(getSymbols.MySQL,  
  user = "jane", password = "secure")  
> getDefaults("getSymbols.MySQL")  
$user  
[1] "'jane'"  
  
$password  
[1] "'secure'"
```

Symbol lookup: `getSymbols`

- Symbol-specific settings with `setSymbolLookup`
- Map symbol to source
- Rename instrument symbols
 - Avoid clashing with other symbol names
 - Avoid creating non-syntactic names

Reading time-series text files

- `read.zoo` is very flexible
- See the vignette “Reading Data in zoo”
 - `vignette("zoo-read", package="zoo")`