Lecture 2 - Addendum

PPOL 670

Other useful notes that will be useful

Dates

Date values as a special data type that contextualize time information. In order to derive month, days, weekdays among other discrete time attributes from a date, the date values must be converted into date objects.

Suppose we have a string vector with three string elements that represent dates. In order to use the dates, we will need to first convert the strings to dates specifying the correct date format:

```
#Before
  vec <- c("10/10/2009", "1/15/2010", "3/4/2009")
  str(vec)
```

```
##  chr [1:3] "10/10/2009" "1/15/2010" "3/4/2009"
```

```
#After
  vec.dates <- as.Date(vec, "%m/%d/%Y")
  str(vec.dates)
```

```
##  Date[1:3], format: "2009-10-10" "2010-01-15" "2009-03-04"
```

Now that the values are in date form, we can extract various pieces of key information. For example, to extract the month, year or day, we can use `format()`. Notice that format returns values as string vectors.

```
#Year
  format(vec.dates, "%Y")
```

```
## [1] "2009" "2010" "2009"
```

```
#Month
  format(vec.dates, "%m")
```

```
## [1] "10" "01" "03"
```

```
#Day
  format(vec.dates, "%d")
```

```
## [1] "10" "15" "04"
```

To extract the day of week, we can use the `weekdays()` function:

```
#weekdays
  weekdays(vec.dates)
```

```
## [1] "Saturday"  "Friday"    "Wednesday"
```

The full frame

Often times in building panel time series, balanced panels are a necessity – that is for each the units of observation (e.g. the panels) there needs to be an equal number of time units. Identifying all observations in the frame is known as seeing the ‘full frame’.This is particularly necessary for tracking which time periods contain information and which do not. Suppose we wanted to have a list of all months in a range of years as derived from the date example above. We can use the `expand.grid()` function to obtain all unique combinations:

```r
#Create vector of years
  min.year <- as.numeric(format(min(vec.dates), "%Y"))
  max.year <- as.numeric(format(max(vec.dates), "%Y"))
  years <- min.year:max.year

#Create vector of unique months
  months <- 1:12

#Create grid
  full.frame <- expand.grid(year = years, month = months)
  head(full.frame)
```

```
##   year month
## 1 2009     1
## 2 2010     1
## 3 2009     2
## 4 2010     2
## 5 2009     3
## 6 2010     3
```

To check which dates appear in which periods, we'll create a data frame containing all months. We'll then merge this `events` data frame with the `full.frame` data frame using the `year` and `month` fields. To identify which months overlap while preserving the full frame using an outer join, we add a variable `flag` to denote the intersections:

```r
  events <- data.frame(year = as.numeric(format(vec.dates, "%Y")),
                       month = as.numeric(format(vec.dates, "%m")),
                       flag = 1)

  overlap <- merge(full.frame, events,
                   by = c("year", "month"),
                   all.x = TRUE, all.y = TRUE)

  head(overlap)
```

```
##   year month flag
## 1 2009     1   NA
## 2 2009     2   NA
## 3 2009     3    1
## 4 2009     4   NA
## 5 2009     5   NA
## 6 2009     6   NA
```