Multiple time series graphs in one plot disregarding empty fields

multiple time series graphs in one plot disregarding empty fields

Question

I have a question that may be simple but have been having difficult times to find a solution for... I have a data for different companies and for different years that looks like this:

IMAGE OF WIDE FORM SPREADSHEET HERE

I would like to draw a kind of time series graph for all the companies in one single graph. The point is that I don't want to have 0 for missing values for the corresponding intervals. The result that I expect will have the dates as the X axis and values as the Y axis. So, for example, the result for the line for CompanyA will be a horizontal line which starts from 2001-02 and ends at 2001-06 at the height of 1000 (as Y value). I would like to visualize the intervals for different companies... I was trying to use ggplot2 in R to draw it but not really successful... Could anybody help me to draw this in R? I have more than 500 rows and more than 180 columns.

Answer

You have multiple issues here that are squarely in the area of "data wrangling". The biggest issue is to impute actual values into your missing value fields. Luckily, the xts time series library contains functions to do this, as well as a function to plot multiple time series, which is your ultimate goal.

However, before we can use those wonderful functions, you will have to do some work transforming your data into an xts object.

First recreating your data above using the method of @aelwan.

```
df <- read.table(text = c("
CompanyA    NA  1000    NA  NA  NA  1000
CompanyB    600 NA  NA  NA  600 NA
CompanyC    NA  5000    NA  5000 NA  NA"),
    header = F)

colnames(df) <- c("CompanyName", "2001-01", "2001-02", "2001-03", "2001-04",
    "2001-05", "2001-06")

df
```

```
##   CompanyName 2001-01 2001-02 2001-03 2001-04 2001-05 2001-06
## 1    CompanyA      NA    1000      NA      NA      NA    1000
## 2    CompanyB     600      NA      NA      NA     600      NA
## 3    CompanyC      NA    5000      NA    5000      NA      NA
```

Your data appears to be in wide format, so I would suggest transposing it to long format. This will require a few steps to retain important information such as column and row names, as well as the class of your data (numeric).

First, Transpose the data frame

```
df_t <- t(df)
```

Now, save the first row, which contains the company names.

```
company_names <- df_t[1,]
```

The transpose process results in an object of class 'matrix'. Drop the first row and make df_t object class data.frame.

```
df_t <- data.frame(df_t[-1, ], stringsAsFactors = FALSE)
```

Add the company names stored in "company_names" back as the column names

```
colnames(df_t) <- company_names
```

Your column data class might have been lost during the transpose as well, so convert all column to class numeric with the sapply function.

```
df_long <- data.frame(sapply(df_t, FUN = as.numeric), row.names = rownames(df_t))

df_long
```

```
##         CompanyA CompanyB CompanyC
## 2001-01       NA      600       NA
## 2001-02     1000       NA     5000
## 2001-03       NA       NA       NA
## 2001-04       NA       NA     5000
## 2001-05       NA      600       NA
## 2001-06     1000       NA       NA
```

Now, convert your brand new `df_long` data.frame into a time series index based `xts` object to access the time series function you need.

```
library(xts)

# convert rownames "2001-01, 2001-02, ..." to yearmon format
rownames(df_long) <- as.yearmon(rownames(df_long), "%Y-%m")

# pass the dates as an index to the xts via the `order.by` arguement.
df_xts <- xts(df_long , order.by = as.yearmon(rownames(df_long)))
```

Finally, we can use the "Last Observation Carried Forward" function, `na.locf` in the xts package to fill in the dates.

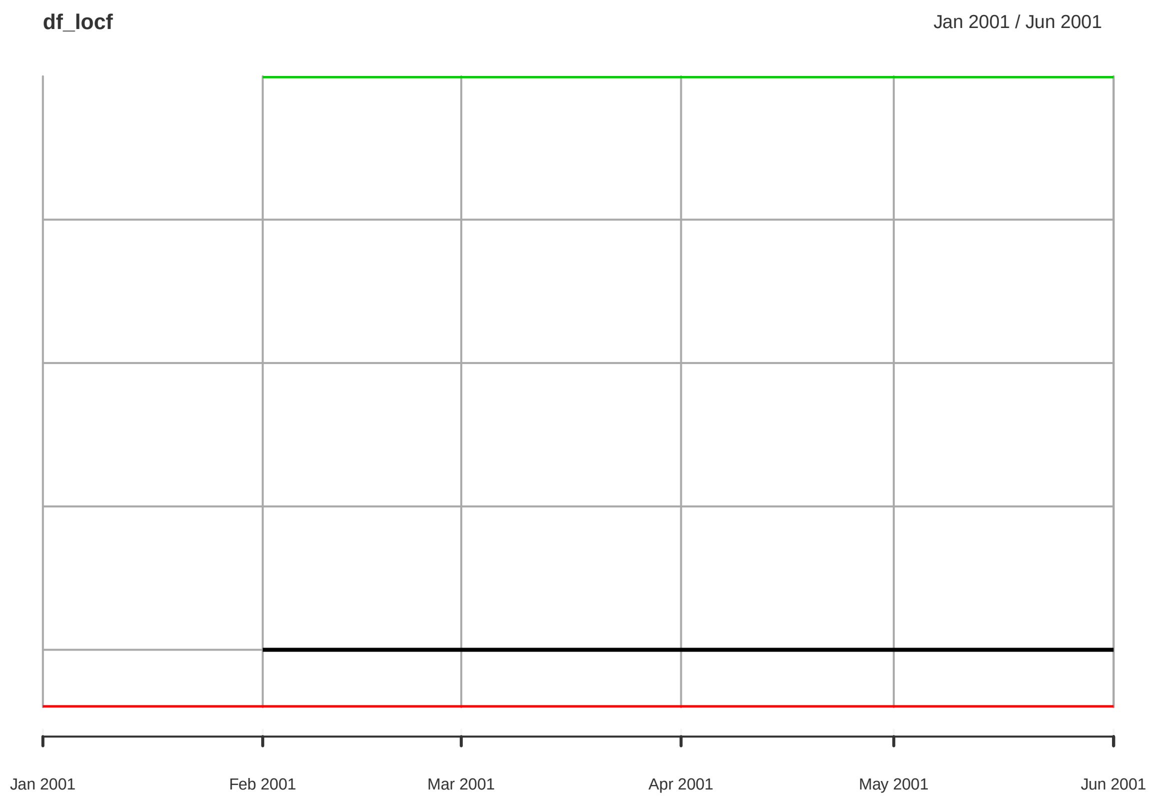
```
df_locf <- na.locf(df_xts)

df_locf
```

```
##          CompanyA CompanyB CompanyC
## Jan 2001       NA      600       NA
## Feb 2001     1000      600     5000
## Mar 2001     1000      600     5000
## Apr 2001     1000      600     5000
## May 2001     1000      600     5000
## Jun 2001     1000      600     5000
```

When calling the `plot` function on objects of class `xts`, multivariate time series plots are produced automatically.

```r
# The plot function works.
plot(df_locf)
```


df_locf
Jan 2001 / Jun 2001
Jan 2001
Feb 2001
Mar 2001
Apr 2001
May 2001
Jun 2001