6 October 2014

TO: RSessions file
FROM: Al Cooper
SUBJECT: Using data.frame objects

What is a data.frame?

A data.frame resembles a matrix or a spreadsheet. For example, it may consists of columns each representing a variable and rows that are the time sequence of observations of that variable. Applied to RAF data files, it may have a structure like this:

Time	ATX	PSXC	WDC	WSC	...
15:00:00	-25.1	410.8	275.4	25.4	...
15:00:01	-25.1	410.9	275.2	25.6	...
15:00:02	-25.2	411.1	275.4	25.5	...
15:00:03	-25.1	411.1	275.1	25.7	...
15:00:04	-25.0	411.2	275.3	25.3	...
...	...	...	...	...	...

All columns must be of the same length, but they may contain different types of variables (Time, character, numeric, logical). Like a spreadsheet, the columns can be assigned names like the header in this table. Rows can also be assigned names, but in the absence of special assignment they will default to the character names '1', '2', '3', '4', ...

Let's get an example. Here is a segment of R code that loads a few selected variables from a netCDF file to a data.frame:

```
require(Ranadu, quietly = TRUE, warn.conflicts=FALSE) # my package of routines

## Loading required package:  RJSONIO
##
## Attaching package:  'signal'
##
## The following objects are masked from 'package:stats':
##
##    filter, poly

Directory <- DataDirectory ()     # for portability; sets the local data directory
Flight <- "rf08"                  # select a flight
Project = "CONTRAST"              # select a project
fname = sprintf("%s%s/%s%s.nc", Directory,Project,Project,Flight)
# XXX set variables needed, here a standard list plus GGVSPDB
```

```
Data <- getNetCDF (fname, standardVariables(c("GGVSPDB")), 60000, 60010)
saveDataFile <- 'RSessionsDataFrame.Rdata.gz'
save (Data, file = saveDataFile, compress='gzip')
N <- names(Data)
```

The resulting names are Time, ATX, DPXC, EWX, GGALT, LATC, LONC, MACHX, MR, PALT, PSXC, QCXC, TASX, WDC, WSC, WIC, GGVSPDB. The data.frame 'Data' looks like this:

```
##                   Time    ATX      DPXC   EWX GGALT  LATC  LONC  MACHX
## 1  2014-02-01 06:00:00  9.664 -1.74589 5.380  3271 14.14 154.3 0.5517
## 2  2014-02-01 06:00:01  9.600 -0.58043 5.860  3258 14.14 154.3 0.5523
## 3  2014-02-01 06:00:02  9.608 -0.10240 6.067  3245 14.14 154.3 0.5524
## 4  2014-02-01 06:00:03  9.649  0.04999 6.135  3231 14.14 154.3 0.5525
## 5  2014-02-01 06:00:04  9.745  0.18771 6.197  3218 14.14 154.3 0.5521
## 6  2014-02-01 06:00:05  9.832  0.30350 6.249  3205 14.14 154.3 0.5522
## 7  2014-02-01 06:00:06  9.898  0.65490 6.410  3191 14.14 154.3 0.5525
## 8  2014-02-01 06:00:07  9.972  0.56748 6.369  3178 14.15 154.3 0.5528
## 9  2014-02-01 06:00:08 10.021  0.50978 6.342  3164 14.15 154.3 0.5536
## 10 2014-02-01 06:00:09 10.103  0.45882 6.319  3151 14.15 154.3 0.5537
## 11 2014-02-01 06:00:10 10.179  0.53403 6.354  3137 14.15 154.3 0.5538
##       MR PALT  PSXC  QCXC  TASX   WDC   WSC     WIC GGVSPDB
## 1  4.866 3091 693.0 159.2 185.9 59.76 10.64 -0.2153  -13.53
## 2  5.295 3078 694.2 159.9 186.1 57.36 10.54 -0.2423  -13.41
## 3  5.475 3064 695.4 160.2 186.2 56.32 10.52 -0.3595  -13.37
## 4  5.528 3052 696.4 160.5 186.2 55.32 10.58 -0.3402  -13.30
## 5  5.575 3039 697.6 160.5 186.1 54.89 10.55 -0.3907  -13.34
## 6  5.613 3027 698.7 160.8 186.2 53.42 10.54 -0.4517  -13.40
## 7  5.750 3015 699.8 161.3 186.3 51.61 10.53 -0.4760  -13.46
## 8  5.704 3001 701.0 161.7 186.4 49.83 10.57 -0.5229  -13.49
## 9  5.671 2989 702.0 162.5 186.7 47.27 10.72 -0.5907  -13.52
## 10 5.640 2976 703.2 162.9 186.8 45.96 10.79 -0.6423  -13.52
## 11 5.662 2963 704.4 163.2 186.8 44.47 10.80 -0.6980  -13.58
```

Working with data.frames

Addressing elements of a data.frame

You can address particular elements using syntax like the following:

```
Data$ATX[5]

## [1] 9.745

Data[5, 2]            # note the [row,column] syntax

## [1] 9.745

Data[5, ]

##                  Time   ATX   DPXC   EWX GGALT  LATC  LONC  MACHX    MR
## 5 2014-02-01 06:00:04 9.745 0.1877 6.197  3218 14.14 154.3 0.5521 5.575
##   PALT  PSXC  QCXC  TASX   WDC   WSC     WIC GGVSPDB
## 5 3039 697.6 160.5 186.1 54.89 10.55 -0.3907  -13.34

Data[5, "ATX"]

## [1] 9.745

Data$ATX

##  [1]  9.664  9.600  9.608  9.649  9.745  9.832  9.898  9.972 10.021 10.103
## [11] 10.179

Data$ATX[getIndex(Data$Time, 60004)]

## [1] 9.745

Data$ATX[Data$Time == as.POSIXct("2014-02-01 6:00:04", tz='UTC')]

## [1] 9.745
```

Creating subsets of a data.frame

New data.frames that contain subsets of original data.frames can be created using logical vectors. For example:

```
Data[Data$TASX > 186.2, ]
```

```
##                   Time    ATX    DPXC   EWX GGALT  LATC  LONC  MACHX    MR
## 4  2014-02-01 06:00:03  9.649 0.04999 6.135  3231 14.14 154.3 0.5525 5.528
## 7  2014-02-01 06:00:06  9.898 0.65490 6.410  3191 14.14 154.3 0.5525 5.750
## 8  2014-02-01 06:00:07  9.972 0.56748 6.369  3178 14.15 154.3 0.5528 5.704
## 9  2014-02-01 06:00:08 10.021 0.50978 6.342  3164 14.15 154.3 0.5536 5.671
## 10 2014-02-01 06:00:09 10.103 0.45882 6.319  3151 14.15 154.3 0.5537 5.640
## 11 2014-02-01 06:00:10 10.179 0.53403 6.354  3137 14.15 154.3 0.5538 5.662
##    PALT  PSXC  QCXC  TASX   WDC   WSC     WIC GGVSPDB
## 4  3052 696.4 160.5 186.2 55.32 10.58 -0.3402  -13.30
## 7  3015 699.8 161.3 186.3 51.61 10.53 -0.4760  -13.46
## 8  3001 701.0 161.7 186.4 49.83 10.57 -0.5229  -13.49
## 9  2989 702.0 162.5 186.7 47.27 10.72 -0.5907  -13.52
## 10 2976 703.2 162.9 186.8 45.96 10.79 -0.6423  -13.52
## 11 2963 704.4 163.2 186.8 44.47 10.80 -0.6980  -13.58
```

```
Data[setRange(Data$Time, 60005, 60008), ]
```

```
##                  Time    ATX   DPXC   EWX GGALT  LATC  LONC  MACHX    MR
## 6 2014-02-01 06:00:05  9.832 0.3035 6.249  3205 14.14 154.3 0.5522 5.613
## 7 2014-02-01 06:00:06  9.898 0.6549 6.410  3191 14.14 154.3 0.5525 5.750
## 8 2014-02-01 06:00:07  9.972 0.5675 6.369  3178 14.15 154.3 0.5528 5.704
## 9 2014-02-01 06:00:08 10.021 0.5098 6.342  3164 14.15 154.3 0.5536 5.671
##   PALT  PSXC  QCXC  TASX   WDC   WSC     WIC GGVSPDB
## 6 3027 698.7 160.8 186.2 53.42 10.54 -0.4517  -13.40
## 7 3015 699.8 161.3 186.3 51.61 10.53 -0.4760  -13.46
## 8 3001 701.0 161.7 186.4 49.83 10.57 -0.5229  -13.49
## 9 2989 702.0 162.5 186.7 47.27 10.72 -0.5907  -13.52
```

Another useful subset is that omitting all missing-variable rows from the data.frame:

```
na.omit(Data)
```

```
##                   Time    ATX     DPXC   EWX GGALT  LATC  LONC  MACHX
## 1  2014-02-01 06:00:00  9.664 -1.74589 5.380  3271 14.14 154.3 0.5517
## 2  2014-02-01 06:00:01  9.600 -0.58043 5.860  3258 14.14 154.3 0.5523
## 3  2014-02-01 06:00:02  9.608 -0.10240 6.067  3245 14.14 154.3 0.5524
## 4  2014-02-01 06:00:03  9.649  0.04999 6.135  3231 14.14 154.3 0.5525
## 5  2014-02-01 06:00:04  9.745  0.18771 6.197  3218 14.14 154.3 0.5521
## 6  2014-02-01 06:00:05  9.832  0.30350 6.249  3205 14.14 154.3 0.5522
```

```
## 7  2014-02-01 06:00:06  9.898  0.65490 6.410  3191 14.14 154.3 0.5525
## 8  2014-02-01 06:00:07  9.972  0.56748 6.369  3178 14.15 154.3 0.5528
## 9  2014-02-01 06:00:08 10.021  0.50978 6.342  3164 14.15 154.3 0.5536
## 10 2014-02-01 06:00:09 10.103  0.45882 6.319  3151 14.15 154.3 0.5537
## 11 2014-02-01 06:00:10 10.179  0.53403 6.354  3137 14.15 154.3 0.5538
##       MR PALT  PSXC  QCXC  TASX   WDC   WSC     WIC GGVSPDB
## 1  4.866 3091 693.0 159.2 185.9 59.76 10.64 -0.2153  -13.53
## 2  5.295 3078 694.2 159.9 186.1 57.36 10.54 -0.2423  -13.41
## 3  5.475 3064 695.4 160.2 186.2 56.32 10.52 -0.3595  -13.37
## 4  5.528 3052 696.4 160.5 186.2 55.32 10.58 -0.3402  -13.30
## 5  5.575 3039 697.6 160.5 186.1 54.89 10.55 -0.3907  -13.34
## 6  5.613 3027 698.7 160.8 186.2 53.42 10.54 -0.4517  -13.40
## 7  5.750 3015 699.8 161.3 186.3 51.61 10.53 -0.4760  -13.46
## 8  5.704 3001 701.0 161.7 186.4 49.83 10.57 -0.5229  -13.49
## 9  5.671 2989 702.0 162.5 186.7 47.27 10.72 -0.5907  -13.52
## 10 5.640 2976 703.2 162.9 186.8 45.96 10.79 -0.6423  -13.52
## 11 5.662 2963 704.4 163.2 186.8 44.47 10.80 -0.6980  -13.58
```

However, be careful using this and other subsetting commands because the time sequence will have gaps and some functions like setRange() won't work, although plots will just skip the missing values. Compare the results from plotWAC (Data$Time, Data$ATX) to D <- na.omit(Data) ; plotWAC (D$Time, D$ATX).

Adding or changing variables in a data.frame

You can operate on variables in the data.frame, changing values, and you can add new variables to the data.frame as follows:

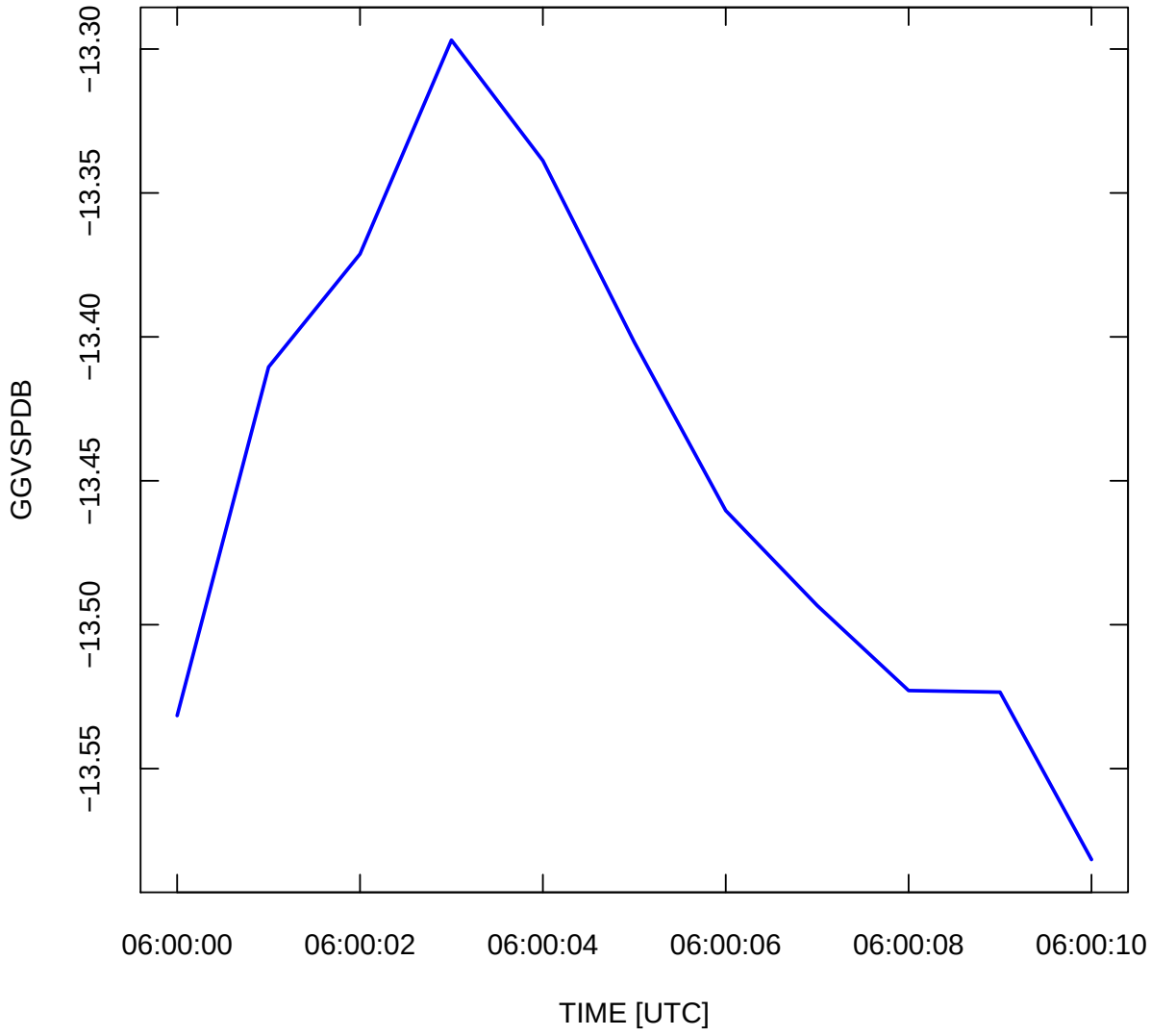
```
# wind component from the east:
Data["UEW"] <- Data$WSC * sin (Data$WDC * pi / 180)
Data$UEW


##  [1] 9.196 8.873 8.751 8.704 8.634 8.464 8.256 8.074 7.877 7.757 7.567
```

Simple plots

Let's plot something:

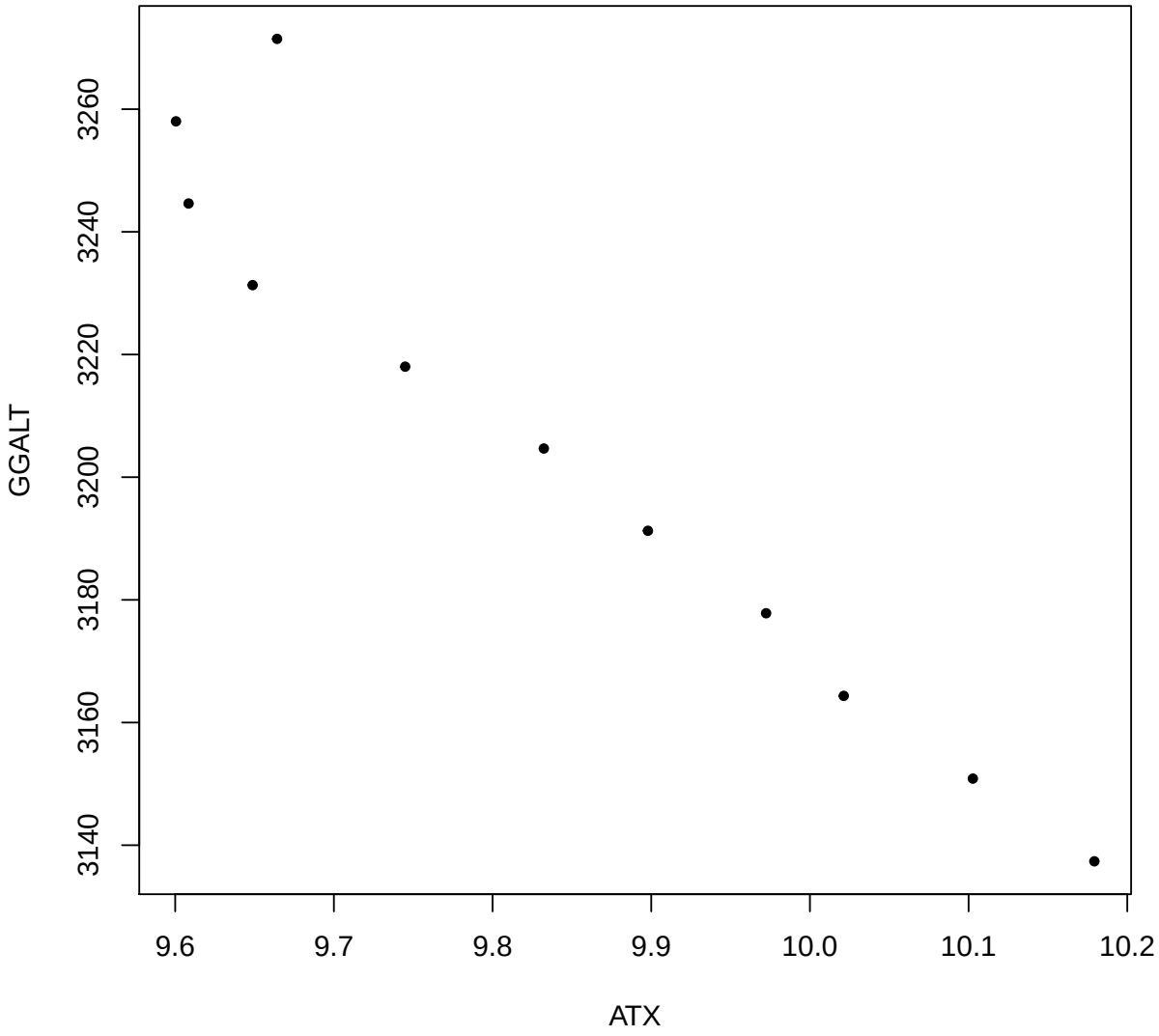
```
plotWAC(Data$Time, Data$GGVSPDB, ylab='GGVSPDB')
```




```
## NULL
```

It is also useful to define special data.frames for constructing plots, especially when using the more advanced plotting capabilities provided by ggplot2. To see a simple scatterplot, you can use the following:

```
D <- Data[, c("ATX", "GGALT")]
plot(D, pch=20)          # pch=20 plots small solid dots
```



Exercise: See what happens if you instead include three variables in the preceding plot.

Exporting to Excel

and now create an Excel spreadsheet with the data:

```
require(xlsx)

## Loading required package:  xlsx
## Loading required package:  rJava
## Loading required package:  xlsxjars

write.xlsx (Data, file="Data.xlsx")
#system("libreoffice Data.xlsx")
```

– End of Memo –

Reproducibility:

PROJECT:	RSessions
ARCHIVE PACKAGE:	RSessionsDataFrame.zip
CONTAINS:	attachment list below
PROGRAM:	/h/eol/cooperw/RStudio/RSessions/RSessionsDataFrame.Rnw
ORIGINAL DATA:	/scr/raf_data/CONTRAST/CONTRASTrf08.nc
GIT:	

Attachments: ProgramFile
Document.pdf
SessionInfo
RSessionsDataFrame.Rdata.gz

```
sink (file="SessionInfo", type="output")
print (sessionInfo ())

## R version 3.1.1 (2014-07-10)
## Platform: x86_64-redhat-linux-gnu (64-bit)
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8          LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8           LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8       LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8          LC_NAME=en_US.UTF-8
##  [9] LC_ADDRESS=en_US.UTF-8        LC_TELEPHONE=en_US.UTF-8
## [11] LC_MEASUREMENT=en_US.UTF-8    LC_IDENTIFICATION=en_US.UTF-8
##
## attached base packages:
```

```
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] xlsx_0.5.7            xlsxjars_0.6.1        rJava_0.9-6
##  [4] Ranadu_0.0-2014-09-30 signal_0.7-4          reshape2_1.4
##  [7] ggthemes_1.7.0        ggplot2_1.0.0         rPython_0.0-5
## [10] RJSONIO_1.3-0         mapdata_2.2-3         mapproj_1.2-2
## [13] maps_2.3-7            nleqslv_2.4           ncdf_1.6.7
## [16] knitr_1.6
##
## loaded via a namespace (and not attached):
##  [1] colorspace_1.2-4 digest_0.6.4     evaluate_0.5.5   formatR_0.10
##  [5] gtable_0.1.2     highr_0.3        MASS_7.3-33      munsell_0.4.2
##  [9] plyr_1.8.1       proto_0.3-10     Rcpp_0.11.2      scales_0.2.4
## [13] stringr_0.6.2    tools_3.1.1


sink ()
system ("zip RSessionsDataFrame.zip RSessionsDataFrame.Rnw RSessionsDataFrame.pdf
        SessionInfo RSessionsDataFrame.Rdata.gz")
```