Exercise II: Lattice Graphics Visualization of Atlantic Hurricanes

Wen-wen Tung

5/23/2019

The US National Hurricane Center and Central Pacific Hurricane Center archive a database of hurricane Best Track Data (HURDAT2). By May 2019, the subset for Atlantic hurricanes, known as the Atlantic HURDAT2 (https://www.nhc.noaa.gov/data/hurdat/hurdat2-format-atlantic.pdf) covers the time period of 1851-2018. The subset for Northeast and North Central Pacific hurricanes the NE/NC Pacific HURDAT2 (https://www.nhc.noaa.gov/data/hurdat/hurdat2-format-nencpac.pdf) has a time range of 1949 to 2017.

Both datasets have a comma-delimited, text format with six-hourly information on the location, maximum winds, central pressure, and (beginning in 2004) size of all known tropical cyclones and subtropical cyclones.

In this exercise, we practice the Atlantic HURDAT2 data visualization using **R**'s `lattice` graphics. The package `lattice` is included with the R Base package in the typical R CRAN (http://cran.r-project.org) distribution. It is an implementation of Trellis Graphics.

References: Landsea, C. W. and J. L. Franklin, 2013: Atlantic Hurricane Database Uncertainty and Presentation of a New Database Format. *Mon. Wea. Rev.*, **141**, 3576-3592.

Sarkar, D, 2008: Lattice: Multivariate Data Visualization with R, Springer. ISBN: 978-0-387-75968-5 http://lmdvr.r-forge.r-project.org/ (http://lmdvr.r-forge.r-project.org/)

Cleveland, W. S., 1993: Visualizing Data, *Hobart Press*, Summit, New Jersey.

Becker, R. A., Cleveland, W. S., and Shyu, M. J., 1996: The Visual Design and Control of Trellis Display, *Journal of Computational and Graphical Statistics*, **5(2)**, 123–155.

1. Atlantic Hurricane Best Track Data

1.1 What's in Atlantic HURDAT2

HURDAT2: http://www.nhc.noaa.gov/data/#hurdat (http://www.nhc.noaa.gov/data/#hurdat)

- Direct Download (can wait): https://www.nhc.noaa.gov/data/hurdat/hurdat2-1851-2018-051019.txt (https://www.nhc.noaa.gov/data/hurdat/hurdat2-1851-2018-051019.txt)
- File has a comma-delimited, text format with essentially six-hourly information of all known tropical cyclones and subtropical cyclones from 1851 to 2016:
  - location in geographic coordinates (degrees)
  - maximum one-minute near-surface (~ 10 m) wind speeds in knots (1 kt = .5144 m/s)
  - minimum central pressure in millibars (1 mb = 1 hPa)
  - size (beginning in 2004).
  - Non-synoptic best track times (other than 00, 06, 12, and 18 Universal Time Coordinate, UTC) mainly to indicate landfalls and intensity maxima
- Codebook http://www.nhc.noaa.gov/data/hurdat/hurdat2-format-atlantic.pdf (http://www.nhc.noaa.gov/data/hurdat/hurdat2-format-atlantic.pdf)

1.2 Load the Atlantic HURDAT2 R dataframe

Option 1: Using the R **HURDAT** package

If you have **R** version 3.4.0 and above and `tidyverse` package completely installed, you could load the Atlantic HURDAT2 data up to 2017 using this option.

- First, install once by typing in your **R** console the command `install.packages("HURDAT")`.
- Then, attach it as a library by typing `library(HURDAT)`.

Option 2: Use the **ALHURDAT.RDATA**

The **R** data object can be downloaded here (https://drive.google.com/file/d/1u5kSYVsod3wCIESFtdYOJXuX9BwJUOGA/view?usp=sharing). The file was created by reading the original raw text file, minimal pre-processing the data, and writing the data into a data frame. This is the faster way, and is therefore used in this exercise. Load it into **R** by **attach("ALHURDAT.RDATA")** or **load("ALHURDAT.RDATA")**.

1.3 Explore Atlantic HURDAT2

A quick look of the original Atlantic HURDAT2 data file, we notice that:

- Cyclones not named are simply referred to as "UNNAMED". We could therefore use the variable "Name" and year information to extract a known storm, such as Katrina (2005) and Sandy (2012).
- Cyclones are referenced by their "Key", a string of alphanumeric characters identifying basin, the number of the storm for the year, followed by the four-digit year.
    - For example, in the key **AL011851**:

– AL = Atlantic Basin ('Basin')

– 01 = First storm of the year ('YearNum')

– 1851 = Year of the storm ('Year')

Question 1: Please `attach` the data file `ALHURDAT.RDATA`. Then, show the objects in the file. (Hint: know your working directory and workspace)

```
#### Answer
```

Question 2: The file contains the data frame `AL`. Please show the structure of `AL` and display summaries of its variables with `str` and `summary` functions.

```
#### Answer
```

Question 3: We have found summary information for following variables. Use the HURDAT codebook (http://www.nhc.noaa.gov/data/hurdat/hurdat2-format-atlantic.pdf) to find out more about their units and physical meaning. Feel free to ask your instructors for more insights. Please note that if you import the data use methods other than attaching `ALHURDAT.RDATA`, your will likely have slightly different output from below.

- Key: Length:50911, Class:character
- Name: Length:50911, Class:character
- DateTime: Length:50911, Class:character
- Record: Length:50911, Class:character
- Status: Length:50911, Class:character
- Lat (numeric)
- Lon (numeric)
- Wind (numeric)
- Pressure (numeric)
- NE34 (numeric)
- and more wind radii (numeric)
- Year (numeric)

Factoring character variables

In the console, type `factor(AL$Key)` , `factor(AL$Name)` , `factor(AL$Record)` , and `factor(AL$Status)` , we will further see that:

- Key has 1864 Levels: AL011851 AL011852 AL011853 AL011854 AL011855 AL011856 AL011857 … AL312005
  - There are 1864 cases of tropical and extra-tropical cyclones
- Name has 289 Levels: ABBY ABLE AGNES ALBERTO ALEX ALFA ALICE ALICIA ALLEN ALLISON ALMA … ZETA
  - Over the North Atlantic Ocean, tropical or subtropical cyclones are named by the National Hurricane Center when they are judged to have intensified into a tropical storms with winds of at least 34 knots (39 mph; 17.49 m/s).
- Record has 9 Levels: C G I L P R S T W
- Status has 10 Levels: DB EX HU LO SD SS TD TS WV, and a mysterious ET not in the codebook

More on the Record and Status identifiers based on HURDAT codebook (http://www.nhc.noaa.gov/data/hurdat/hurdat2-format-atlantic.pdf)

Record identifiers:

- C Closest approach to a coast, not followed by a landfall
- G Genesis
- I An intensity peak in terms of both pressure and wind
- L Landfall (center of system crossing a coastline)
- P Minimum in central pressure
- R Provides additional detail on the intensity of the cyclone when rapid changes are underway
- S Change of status of the system
- T Provides additional detail on the track (position) of the cyclone
- W Maximum sustained wind speed

Status of system:

- TD Tropical cyclone of tropical depression intensity (< 34 knots)
- TS Tropical cyclone of tropical storm intensity (34-63 knots)
- **HU Tropical cyclone of hurricane intensity (> 64 knots)**
- EX Extra-tropical cyclone (of any intensity)
- SD Subtropical cyclone of subtropical depression intensity (< 34 knots)
- SS Subtropical cyclone of subtropical storm intensity (> 34 knots)

- LO A low that is neither a tropical cyclone, a subtropical cyclone, nor an extra-tropical cyclone (of any intensity)
- WV Tropical Wave (of any intensity)
- DB Disturbance (of any intensity)

1.4 Subsetting the data frame

For your convenience, a column of `Year` has been added to `AL` when generating `ALHURDAT.RDATA`. Following code block shows you how it could be done.

```
#### Extract the Year characters from DateTime, then convert to numeric.
#### This step has been performed for ALHURDAT.RDATA, so you don't have to.
AL[, "Year"] <- as.numeric(format(as.Date(AL$DateTime), "%Y"))
```

Now, we can use `Name` and `Year` to extract a specific storm, such as Katrina (2005):

```
#### Extract a subset for Katrina (2005)
Kt <- subset(AL, Year==2005 & Name == "KATRINA")
```

Question 4: Explore the structure and summarize the variables in the Katrina data subset `Kt`.

```
#### Answer
```

Question 5: Are you able to find out which storm has the mysterious “ET” status? Can you make a conjecture of what the status means?

```
#### Answer
```

2. Visualizing Atlantic HURDAT2

2.1 Introduction to Trellis Display via **R**’s Lattice Graphics.

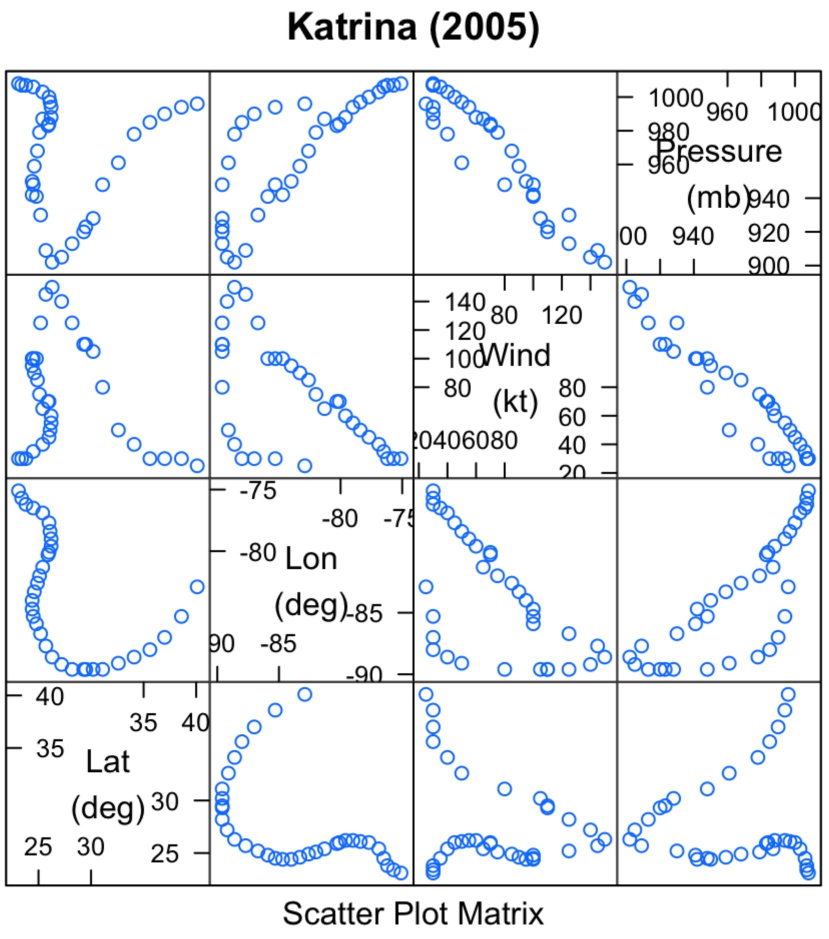
Load the `lattice` package

```
library(lattice)
```

2.1.1 Scatter plot matrices `splom()`

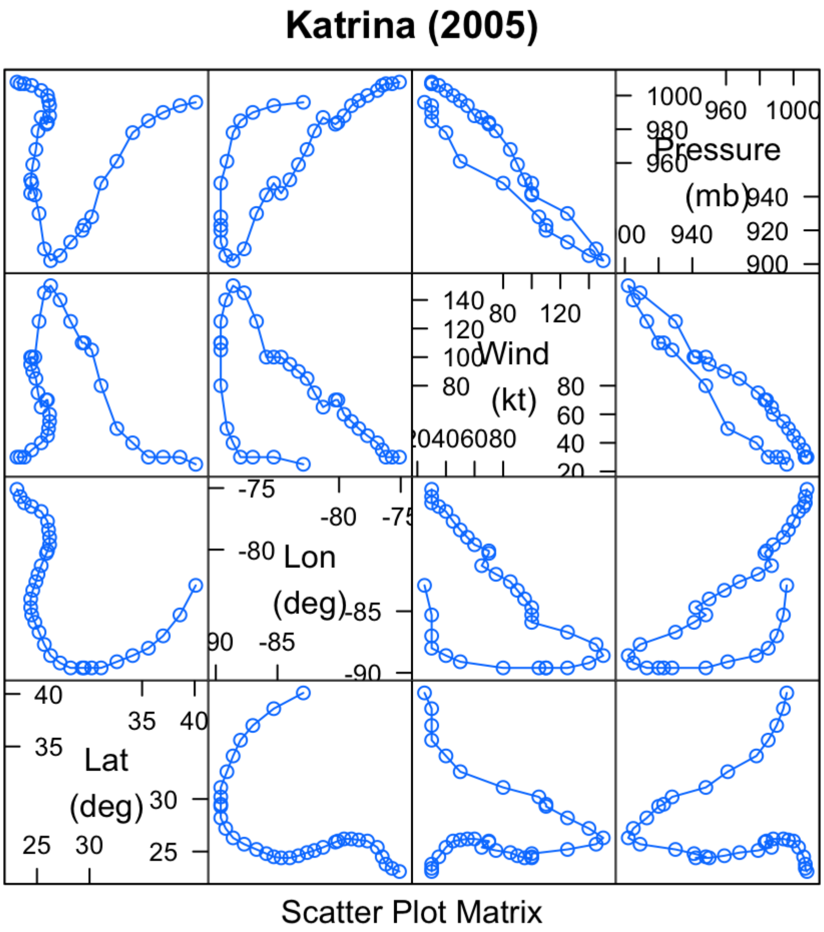
Explore some variables in the Katrina (2005) subset. Example 1:

```
splom(~Kt[,c(6:9)],
      varnames = c("Lat\n(deg)",
                   "Lon\n(deg)",
                   "Wind\n(kt)",
                   "Pressure\n(mb)"),
      main="Katrina (2005)"
      )
```



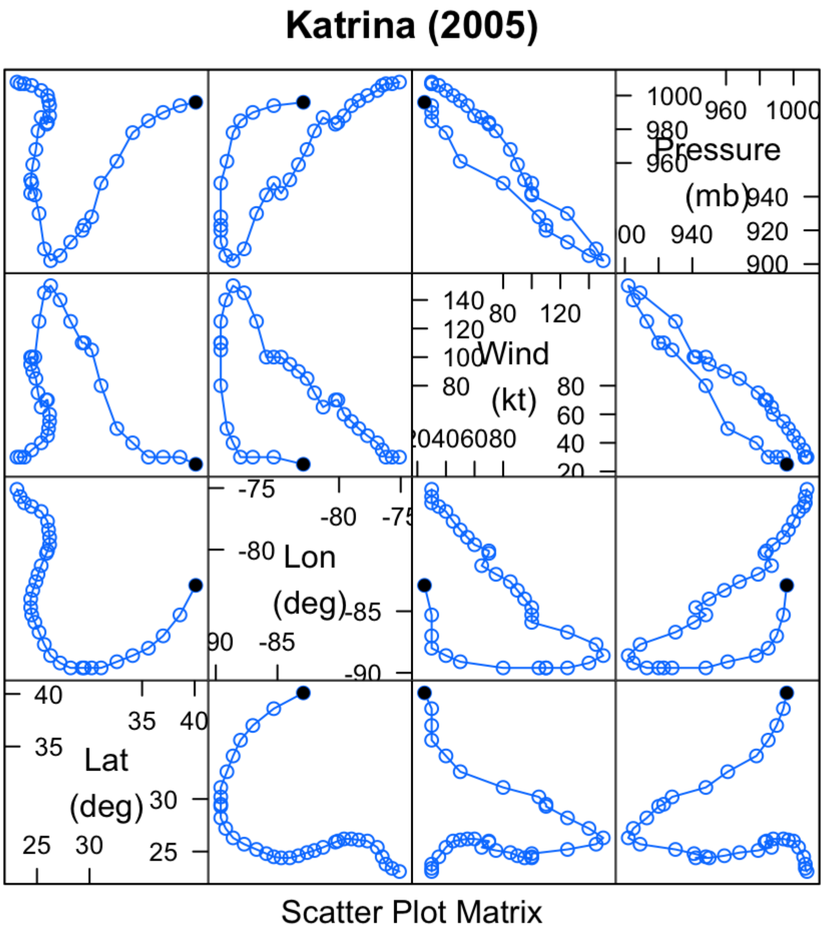
splom() Example 2:

```
# type = "b" means "both" lines and points
splom(~Kt[,c(6:9)],
      varnames = c("Lat\n(deg)",
                   "Lon\n(deg)",
                   "Wind\n(kt)",
                   "Pressure\n(mb)"),
      type="b",
      main="Katrina (2005)"
      )
```



`splom()` Example 3:

```
# Fill the last data points with color black
nstep <- dim(Kt)[1]
splom(~Kt[,c(6:9)],
      varnames = c("Lat\n(deg)",
                   "Lon\n(deg)",
                   "Wind\n(kt)",
                   "Pressure\n(mb)"),
      type="b",
      main="Katrina (2005)",
      panel = function(x,y,...){
        panel.splom(x,y,...)
        panel.xyplot(x[nstep],y[nstep], pch = 16,
                     col="black")
        }
      )
```



Question 6: Following the previous examples, first extract Sandy (2012) and save it as a new data frame `Sd`. Then, apply `splom`. In addition to the black filled last data point, fill the first data point with color red.

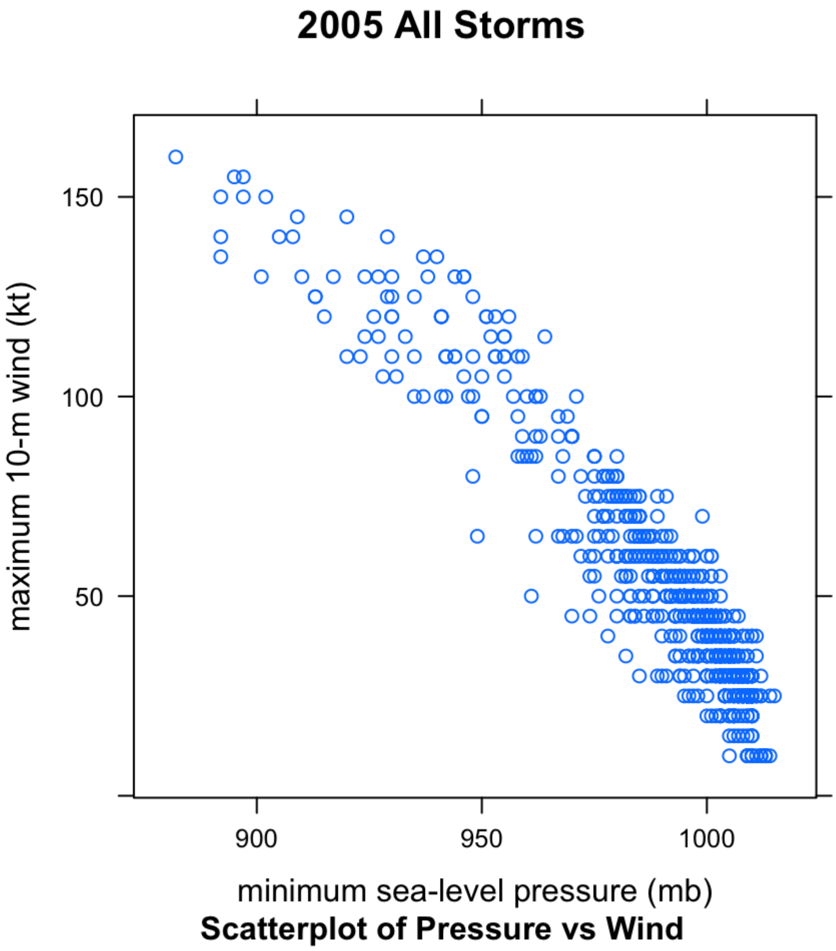
```
#### Answer
```

2.1.2 Common bivariate scatterpots `xyplot()`

There appears to be some inverse relation between `Wind` and `Pressure` (maximum wind and minimum pressure near the surface) in the cases we observed thus far.

Therefore, we plot `Wind` vs. `Pressure` for Year 2005

```
xyplot(Wind ~ Pressure,
       data = AL,
       subset = Year == 2005,
       aspect = 1,
       xlab="minimum sea-level pressure (mb)",
       ylab="maximum 10-m wind (kt)",
       main="2005 All Storms",
       sub="Scatterplot of Pressure vs Wind")
```



Question 7: Subset `Wind` and `Pressure` data for Year 2012 and create a bivariate scatterplot.

```
#### Answer
```

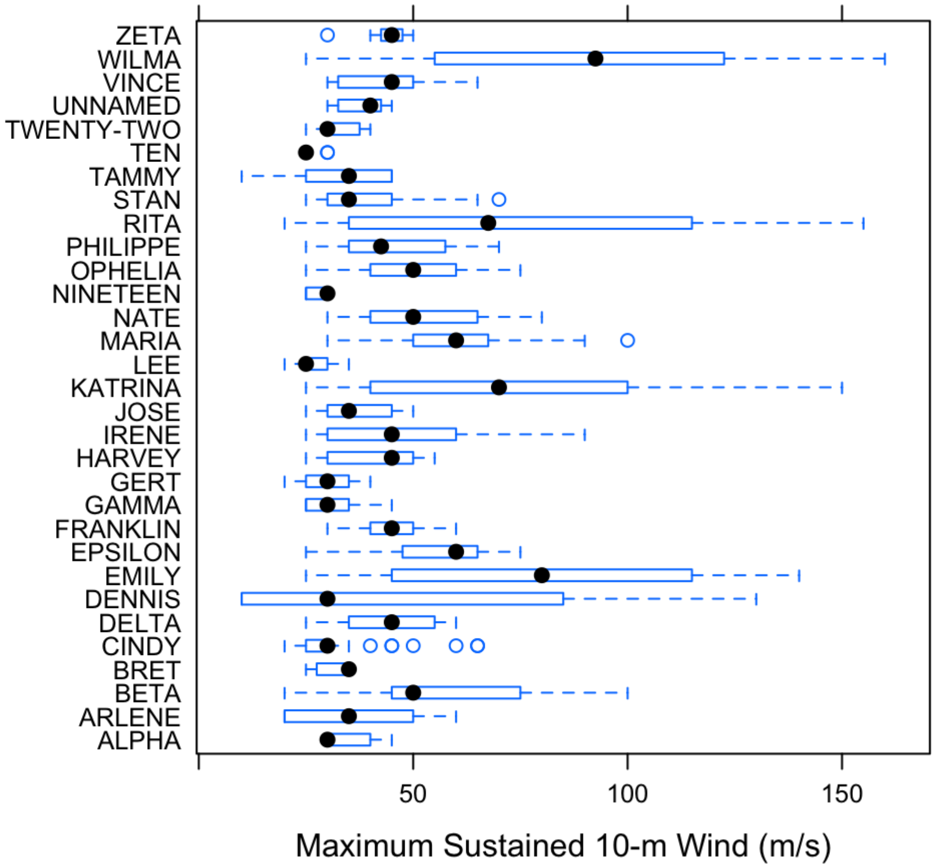
2.1.3 The box-and-whisker plots `bwplot()`

Box and whisker plots summarize the distribution of a variable using a few quantiles. Here, we create box-and-whisker plots of the maximum sustained 10-m wind observations for each storm in 2005. Note that in **R**, even a figure is an object. As in the example blow, we could assign the plot to `x`. The class of `x` is “trellis”, meaning it was generated using Trellis Graphics. We could print or edit `x` later.

```
x <- bwplot(Name ~ Wind,
       data = AL,
       subset = Year == 2005,
       aspect = 1,
       NA.RM=TRUE,
       xlab="Maximum Sustained 10-m Wind (m/s)")
class(x)
```

```
## [1] "trellis"
```

```
print(x)
```



By default the whiskers produced by `bwplot()` extend to the most extreme data points no more than 1.5 times the length of the box away from the box, and any data points outside that range are marked as potential outliers.

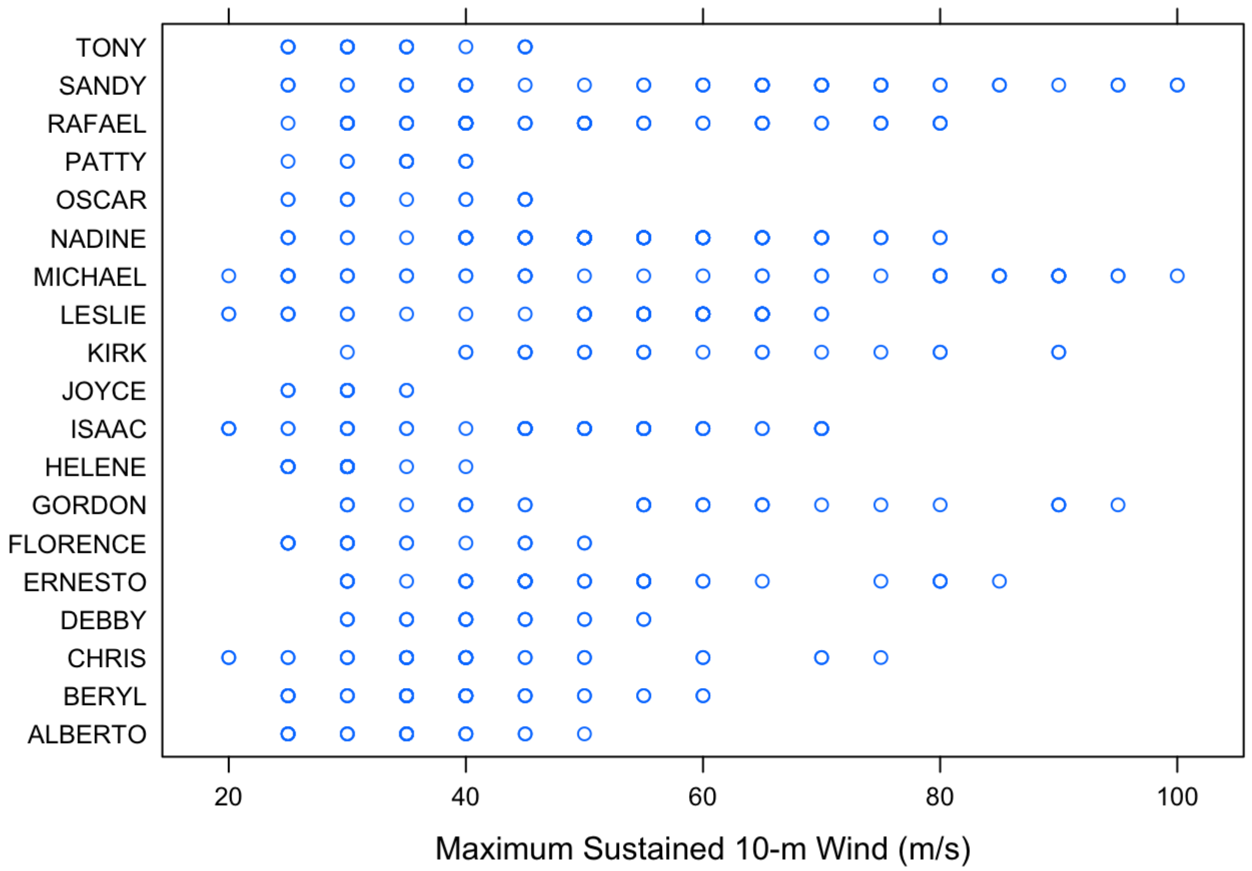
Question 8: Similar to the example, create a box-and-whisker plot for Year 2012. Describe what you observe.

```
#### Answer
```

2.1.4 One-dimensional scatterplots `stripplot()`

In the following, we create a 1-D scatterplot of maximum sustained 10-m wind speed for each named storm in 2012 using `stripplot`.

```
# Create stripplot for Year 2012
stripplot(Name ~ Wind,
       data = AL,
       subset = Year == 2012,
       jitter = FALSE,
       xlab="Maximum Sustained 10-m Wind (m/s)")
```



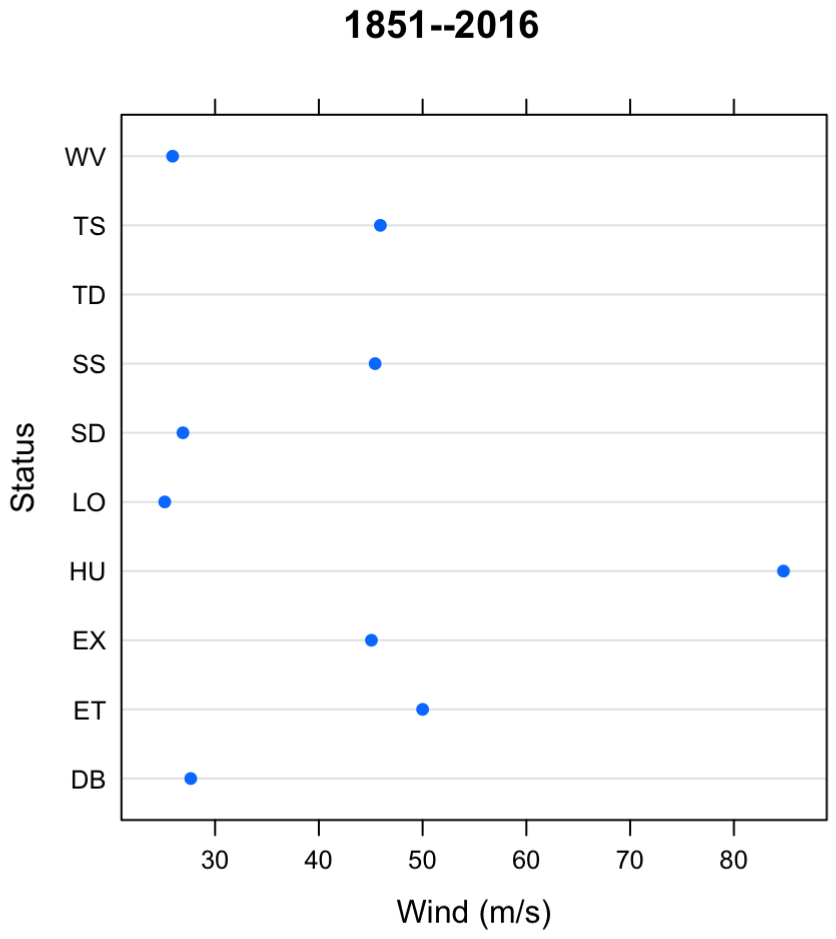
Question 9: Jitter the data points by changing `jitter` to `TRUE` . Describe what you see and what you think jittering the data help here?

```
#### Answer
```

2.1.5 Cleveland dot plots `dotplot()`

Let us exploit the Status of storms. Find out the mean of Wind of each Status throughout the record. The code also demonstrates the use of one of **R**'s “apply” function, `tapply` . It applies the operation of taking mean value to the `Wind` values grouped according to the storm `Status` .

```
status.mean <- tapply(AL$Wind, AL$Status, mean)
dotplot( names(status.mean)~status.mean,
aspect = 1,
main = "1851--2016",
ylab = "Status",
xlab = "Wind (m/s)" )
```



Question 10: Did you notice something wrong with the mean maximum sustained 10-m wind for `TD`? What could have been wrong? How do you fix it?

```
#### Answer
```

Question 11: Following code cell is missing critial parts. Complete it to find out the mean of minimum central pressure, `Pressure`, of each `Status` from 1990 and beyond.

```
#### Answer
statusP.mean <- tapply(
  subset(AL,Year>= )$Pressure,
  subset(AL,Year>= )$Status, mean)
dotplot( ,
aspect = 1,
main = "1990 -- 2018",
ylab = "Status",
xlab = "Pressure (mb)"
)
```

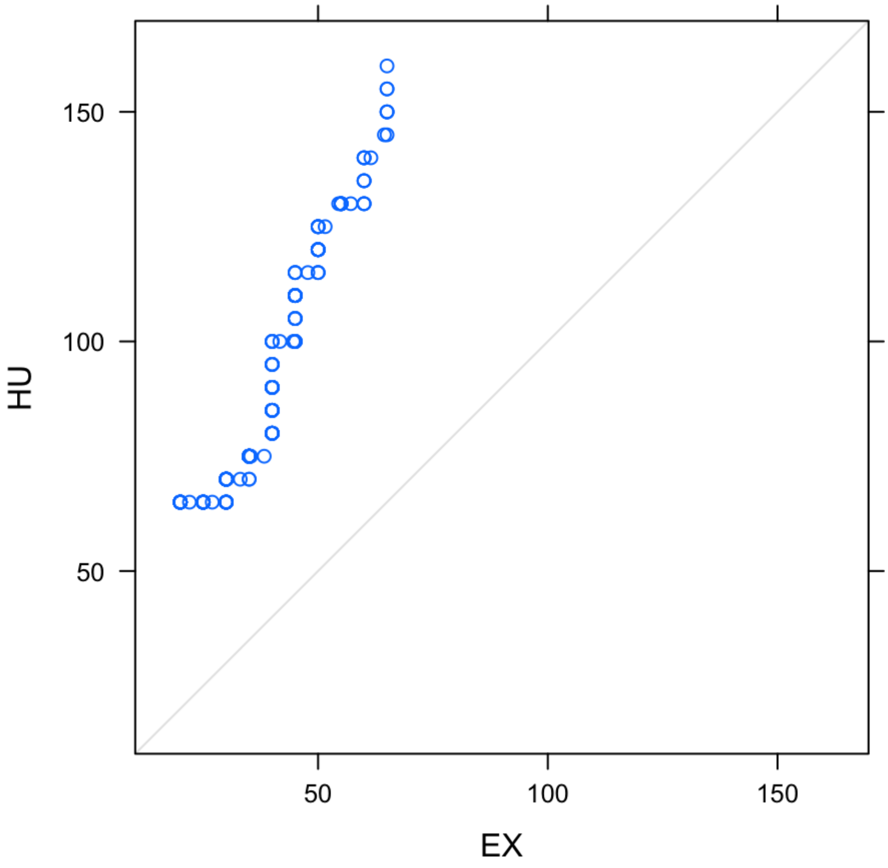
2.1.6 Quantile-Quantile Plots

`qq()`

The function `qq` displays quantile-quantile plots for comparing two distributions.

- Recall these Status identifiers:
  - TS Tropical cyclone of tropical storm intensity (34-63 knots)
  - HU Tropical cyclone of hurricane intensity (> 64 knots)
  - EX Extra-tropical cyclone (of any intensity)
  - SS Subtropical cyclone of subtropical storm intensity (> 34 knots)
- In the following, we can compare the quantiles of HU- and EX-strength maximum sustained 10-m winds in 2005

```
qq(Status ~ Wind,
data = subset(AL, Year==2005 ),
subset = Status=="HU" | Status=="EX",
aspect = 1
)
```



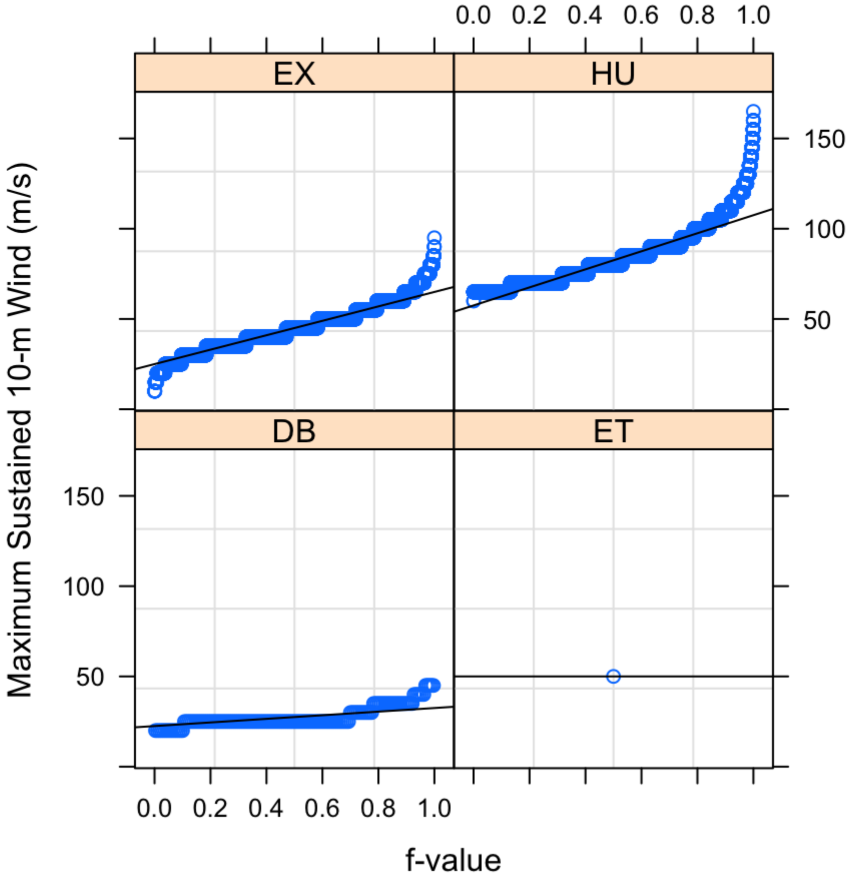
Question 12: Compare the quantiles of HU- and TS-strength Winds in 2005. Comment the plot and compare it with the previous plot.

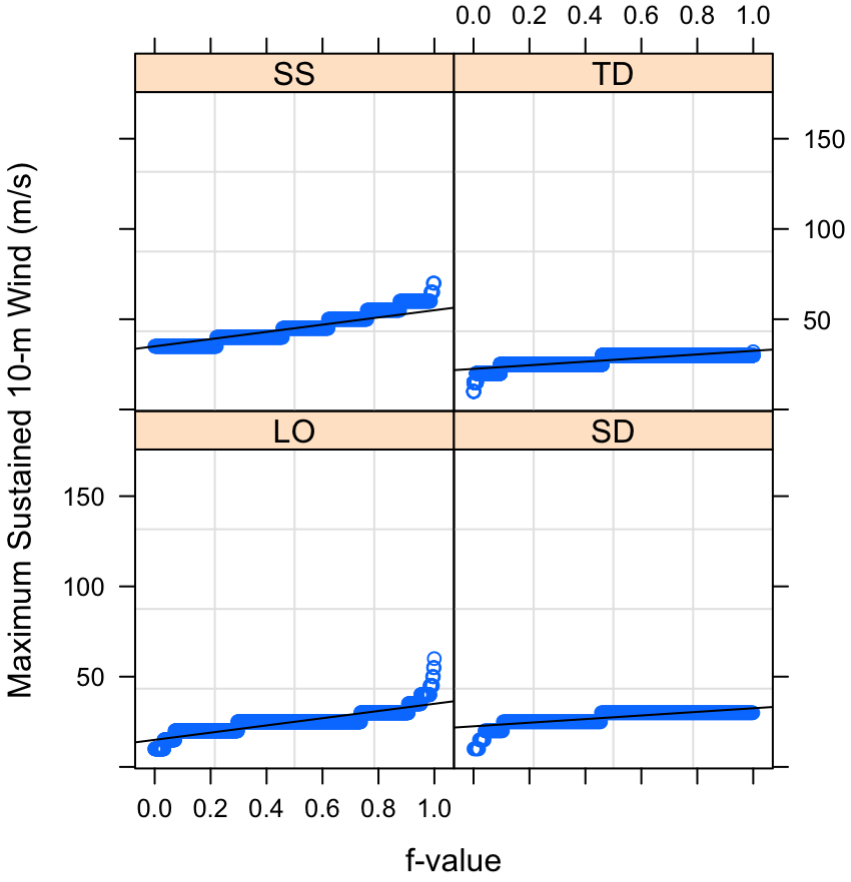
```
#### Answer
```

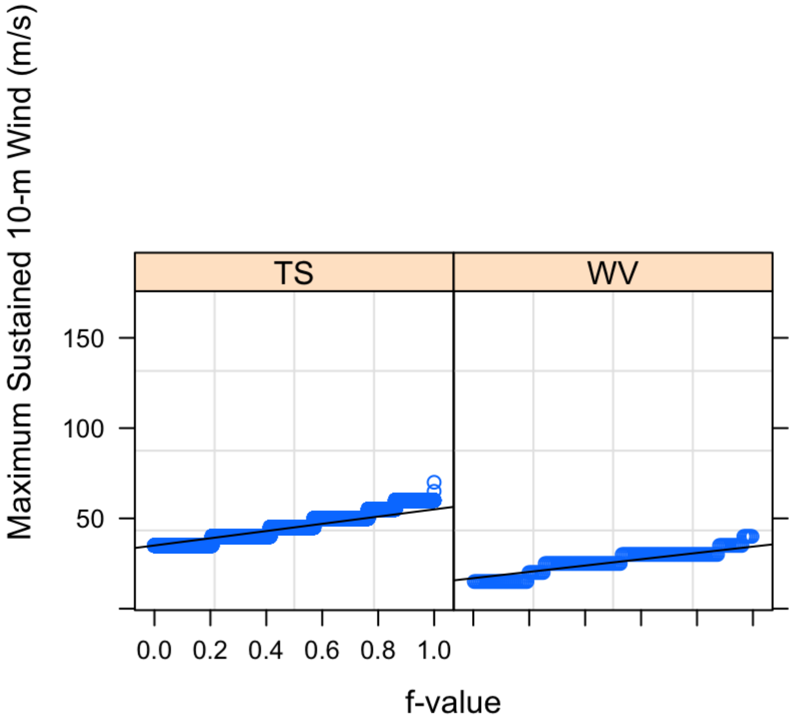
`qqmath()`

To compare a sample distribution against a theoretical distribution, we use `qqmath` . In the following code, we compare the maximum sustained 10-m wind, conditioned on the storm status, against the theoretical uniform distribution for all years.

```
qqmath(~Wind | Status,
distribution = qunif,
data = AL,
panel = function(x, ...) {
panel.grid()
panel.qqmath(x,...)
panel.qqmathline(x, probs = c(0.25, 0.75),...)
},
layout = c(2,2),
aspect = 1,
xlab = "f-value",
ylab = "Maximum Sustained 10-m Wind (m/s)")
```


EX
HU
DB
ET
Maximum Sustained 10-m Wind (m/s)
f-value
0.0
0.2
0.4
0.6
0.8
1.0
50
100
150


SS
TD
LO
SD
Maximum Sustained 10-m Wind (m/s)
f-value
0.0
0.2
0.4
0.6
0.8
1.0
50
100
150

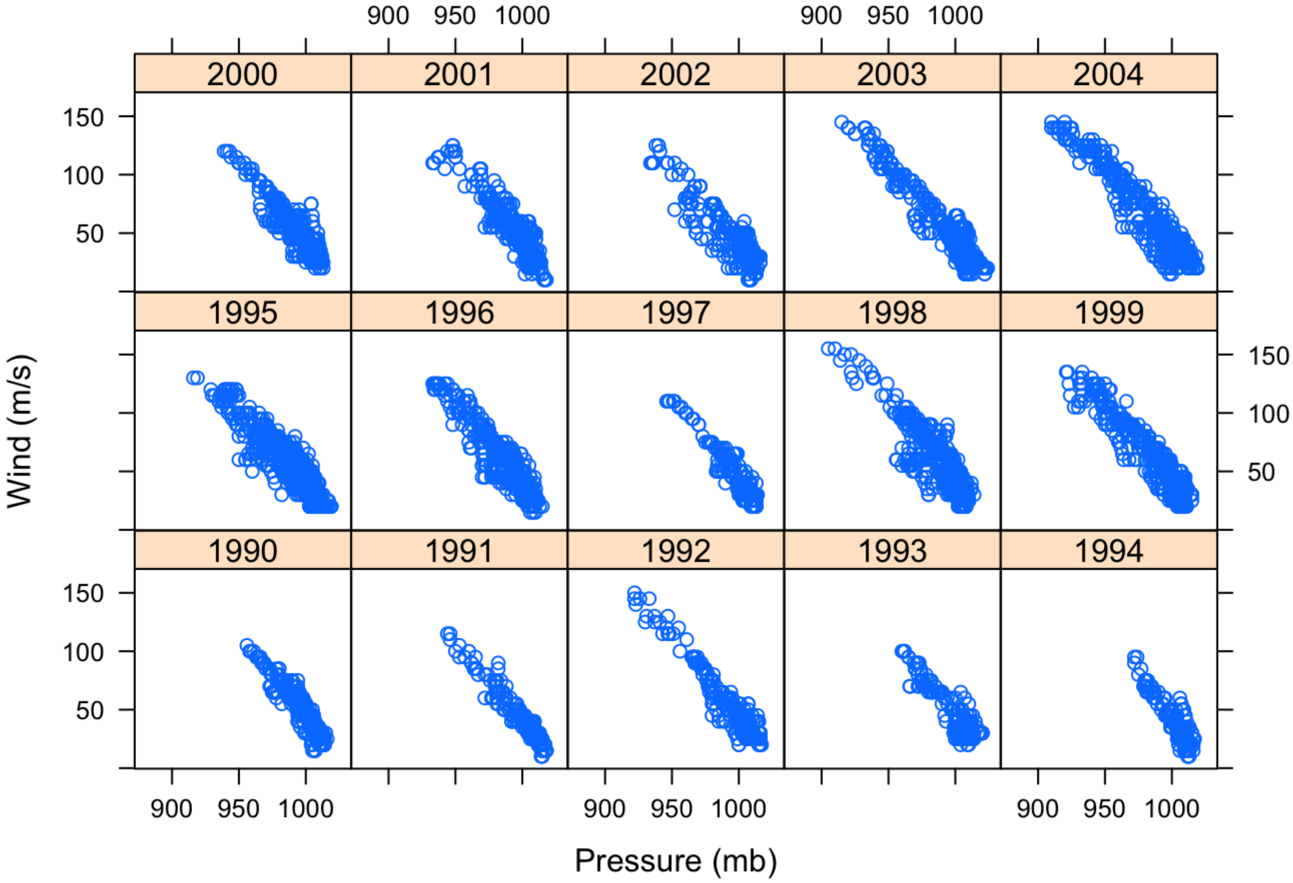


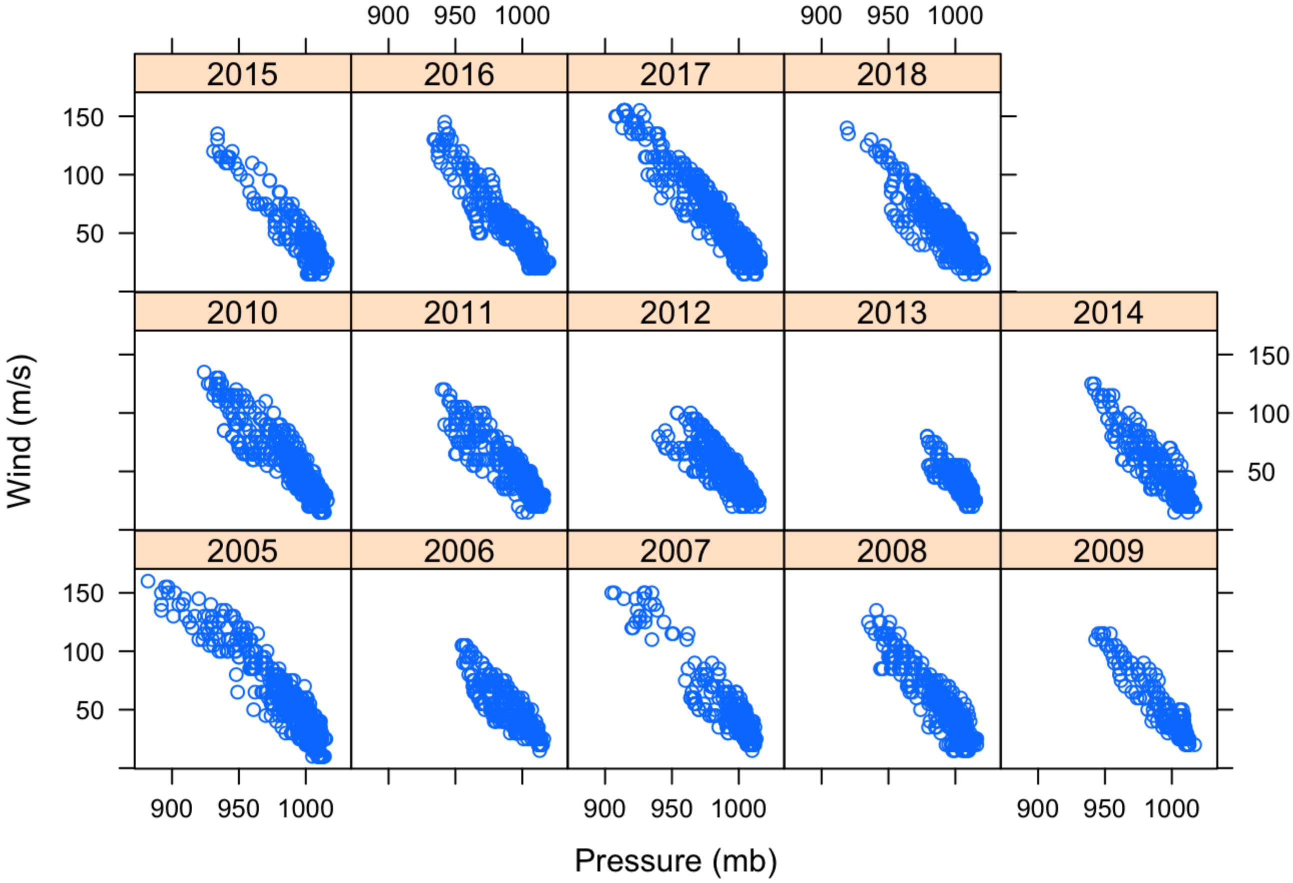
2.2 Multiple panel displays

Trellis Display implemented in `lattice` has a powerful multiple panel display feature unmatched by `ggplot2` . It renders faster and can extend to multiple pages, therefore very suitable for detailed visualization of big data.

Example 1: Multiple `xyplot` with fixed scales to show `Wind` vs `Pressure` from year 1990 to beyond.

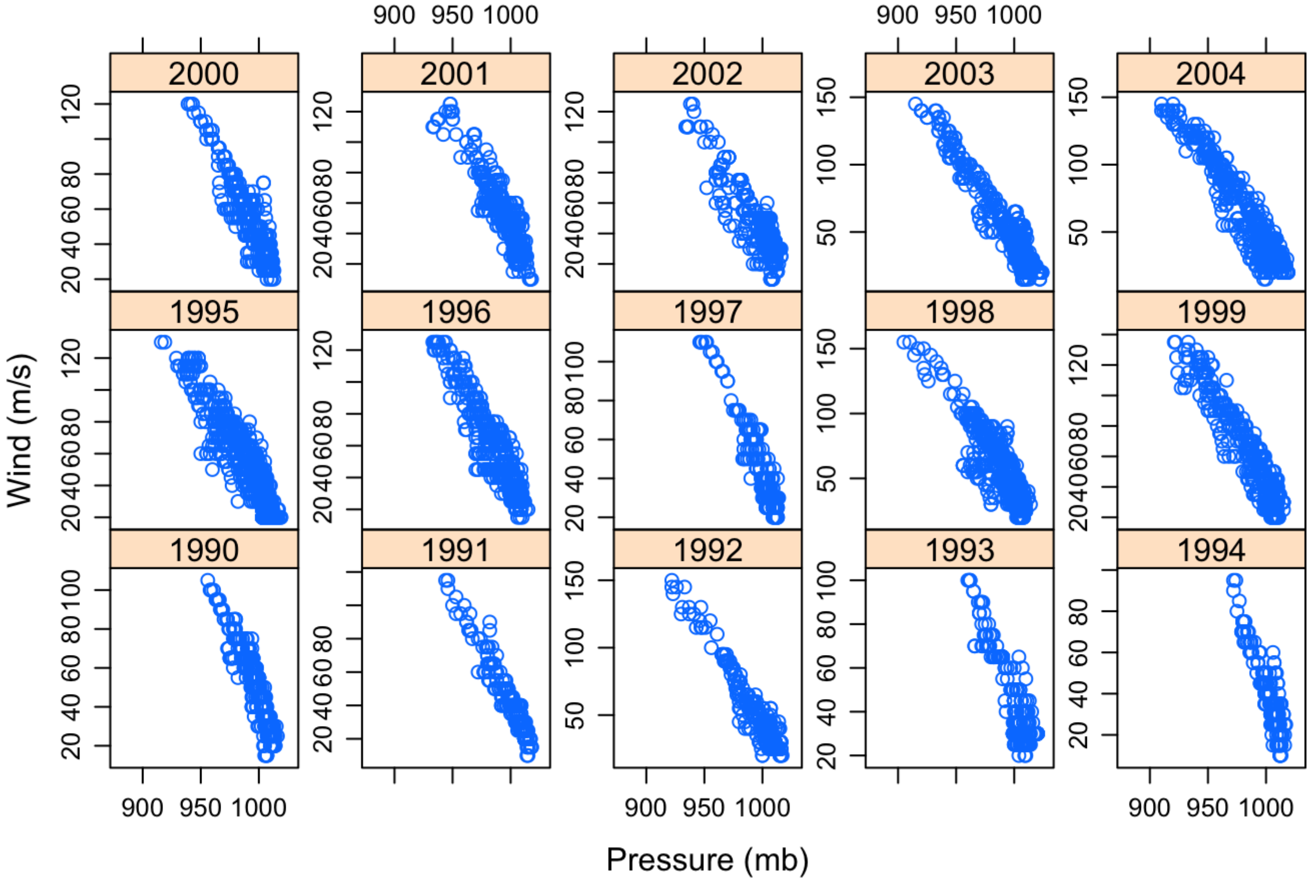
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

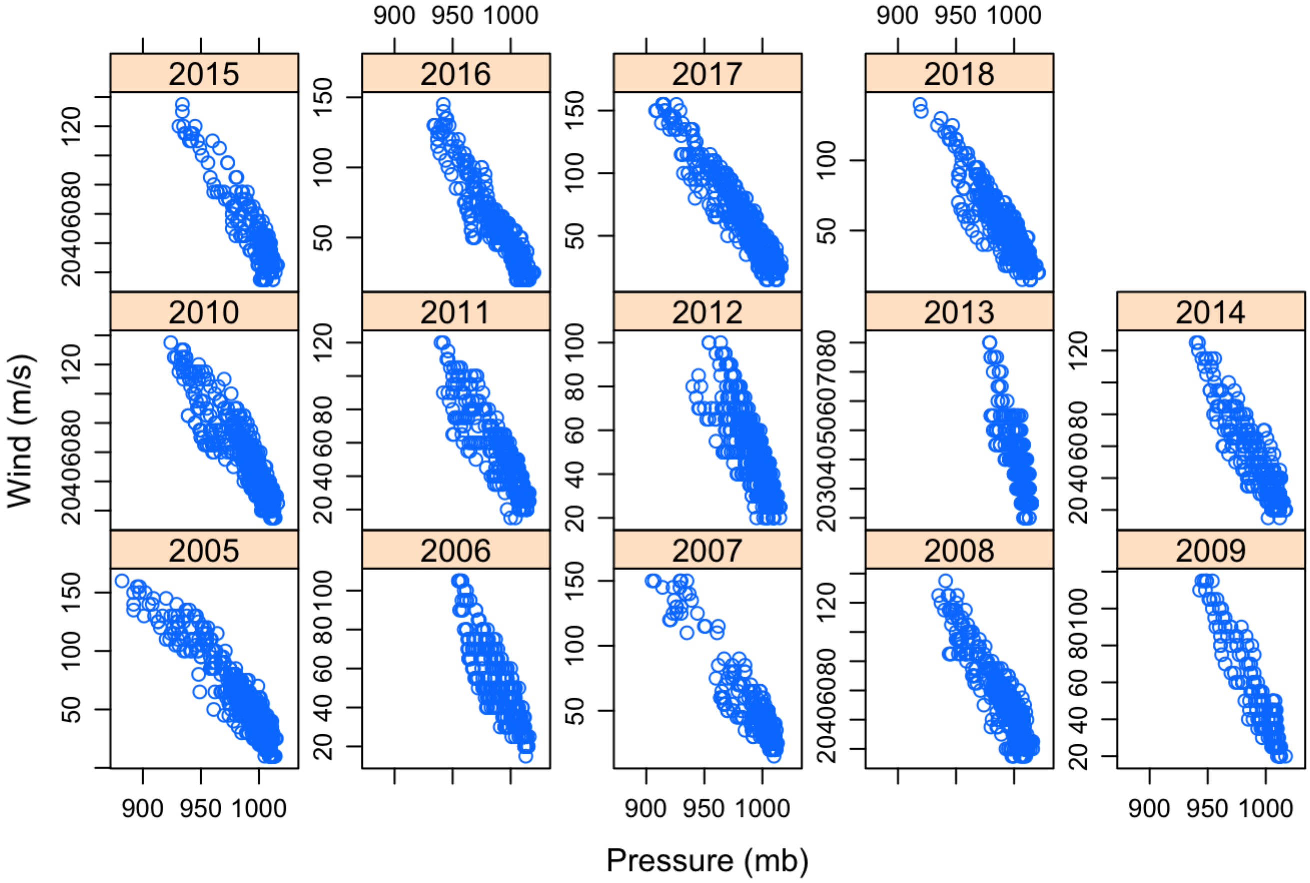




Example 2: Multiple `xyplot` with free y-scales:

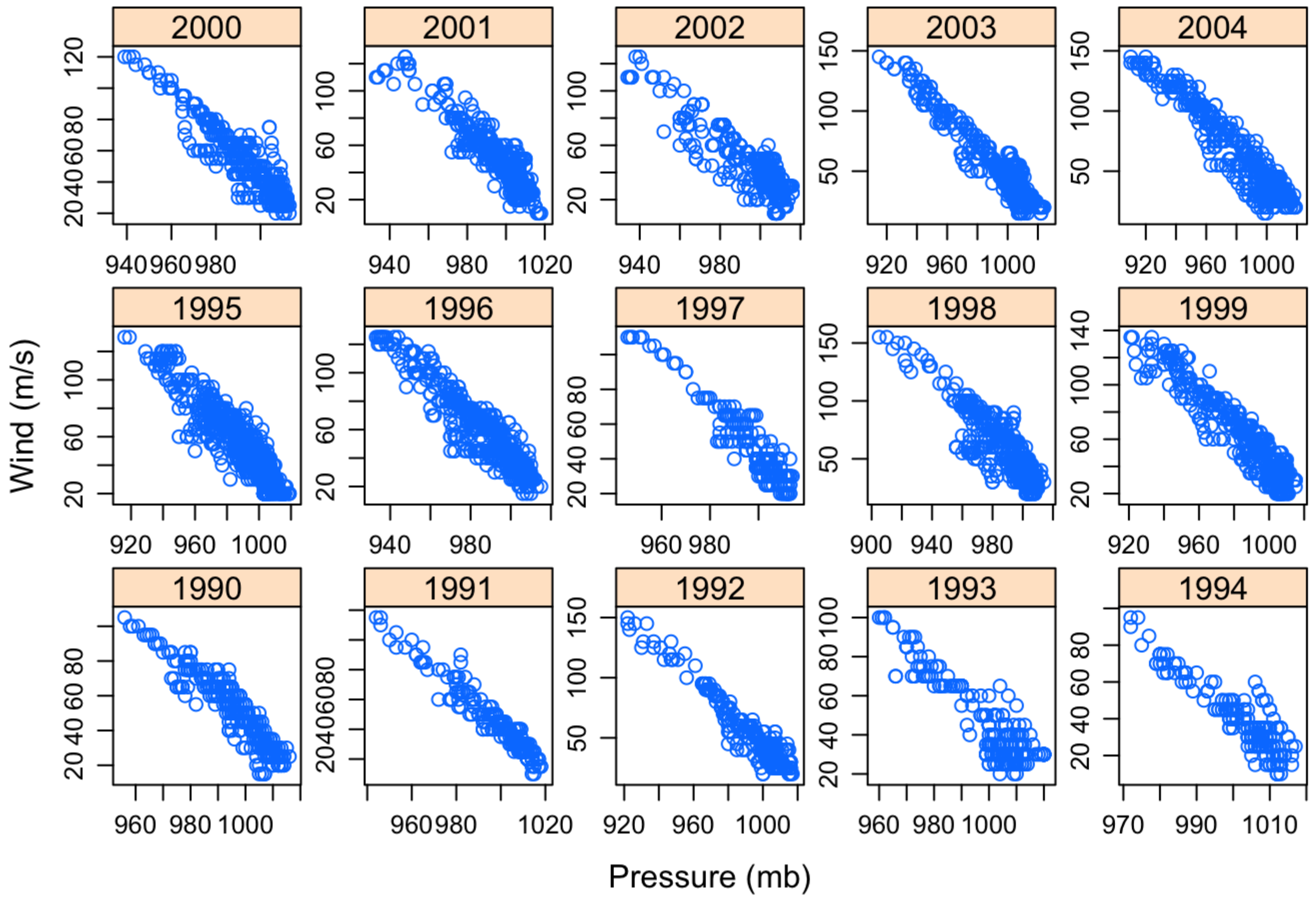
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       scales = list(y="free"),
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

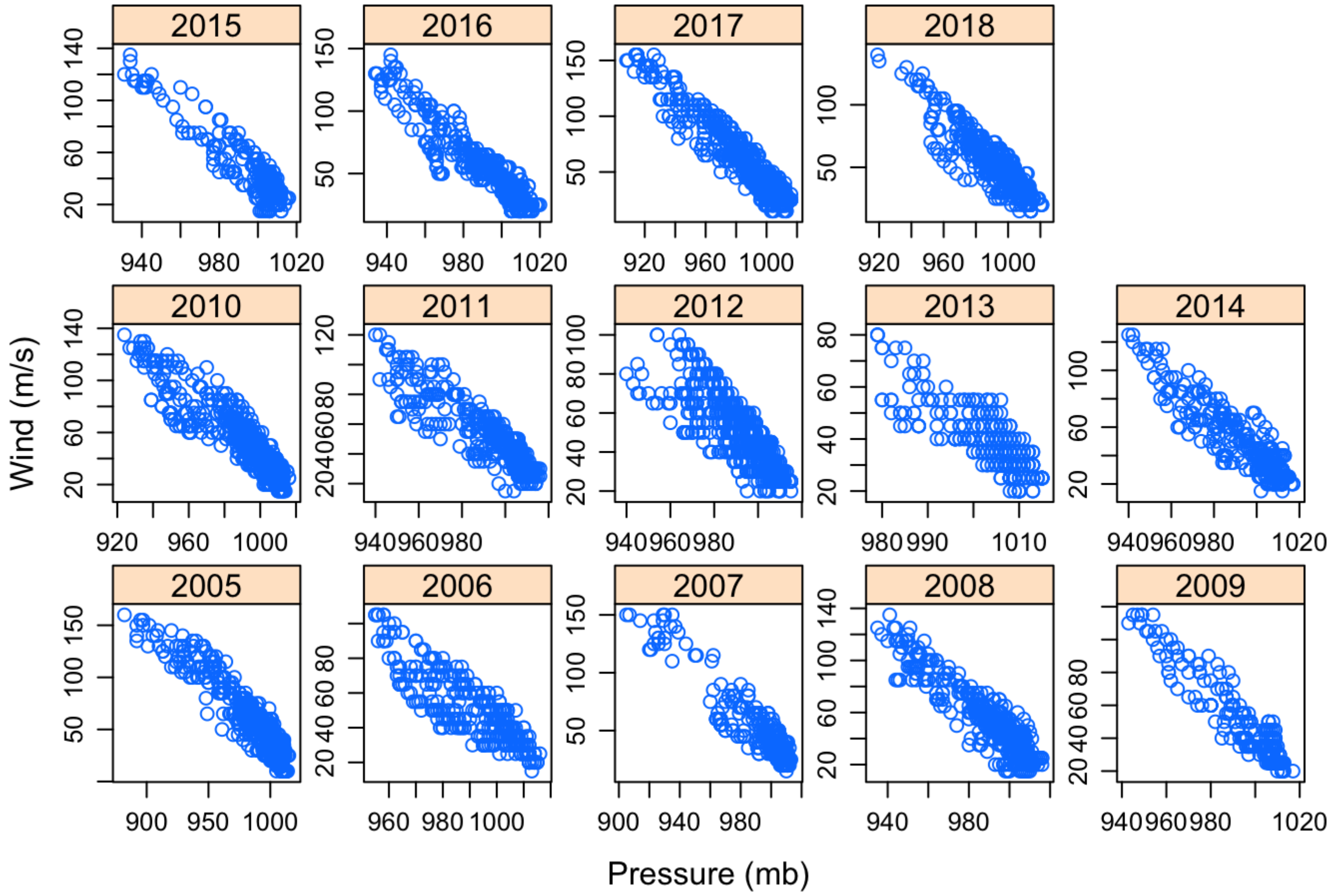




Example 3: Multiple `xyplot` with free x- and y-scales:

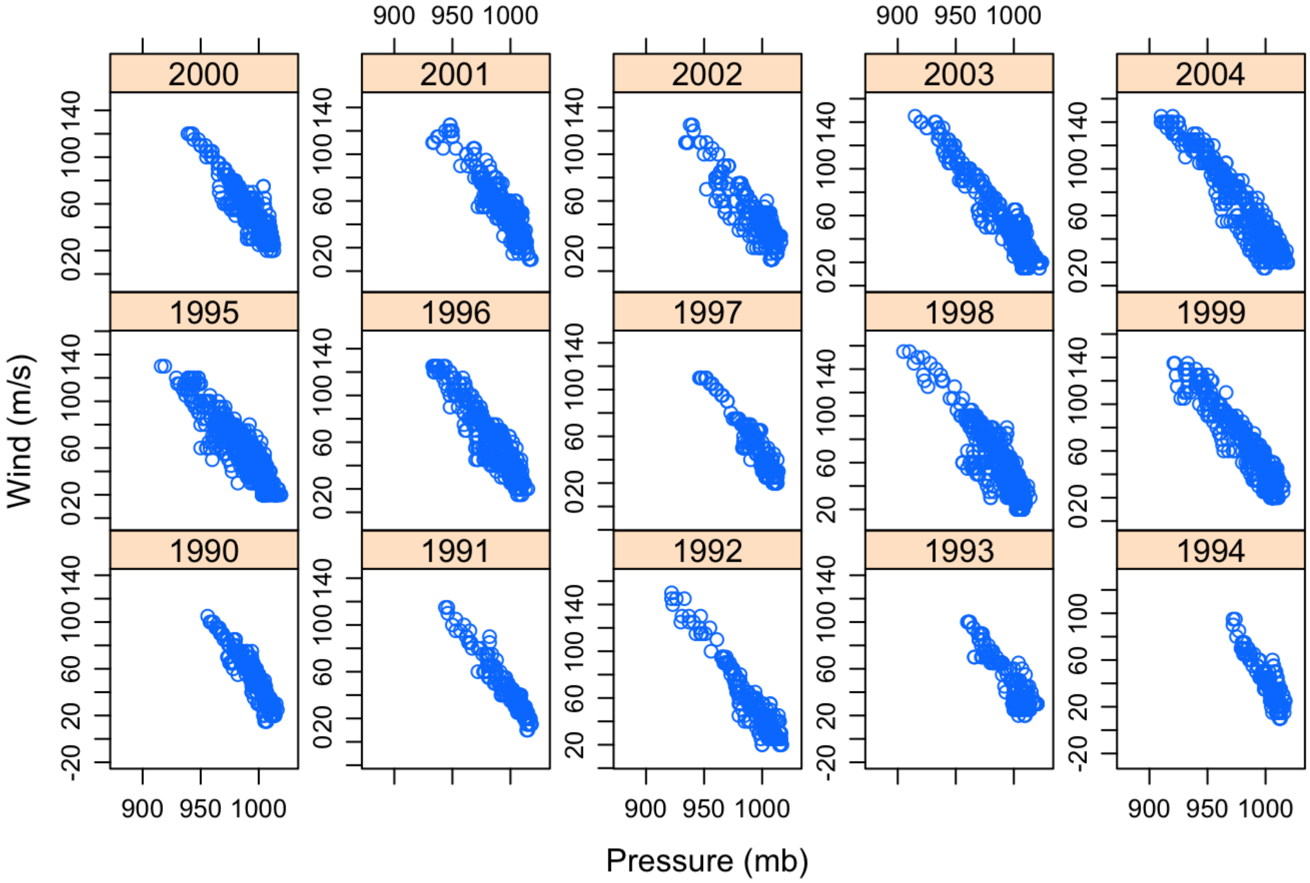
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       scales = list(y="free", x="free"),
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

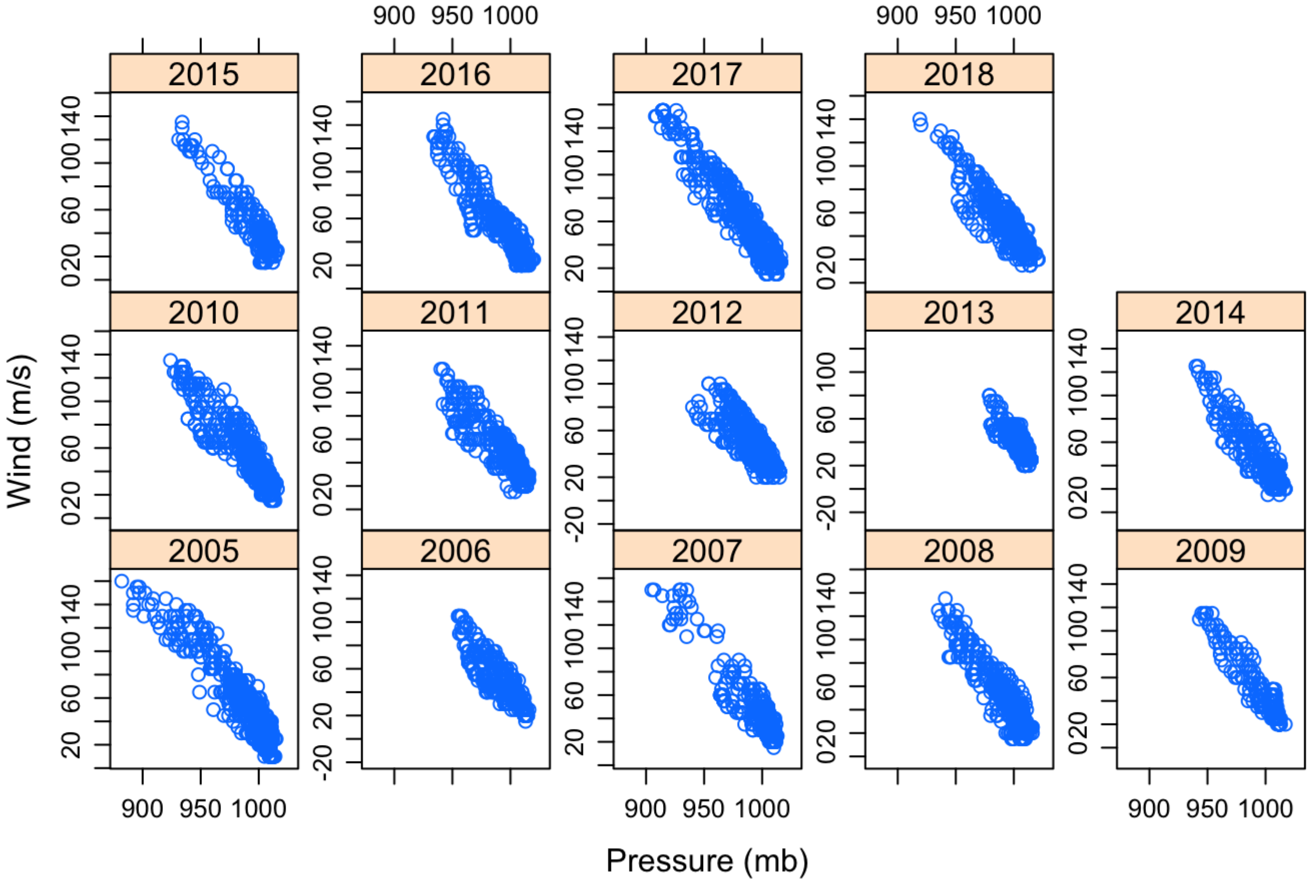




Example 4: Multiple `xyplot` with sliced y-scales and 8 ticks:

```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       scales = list(y=list(tick.number=8,
                             relation="sliced")),
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

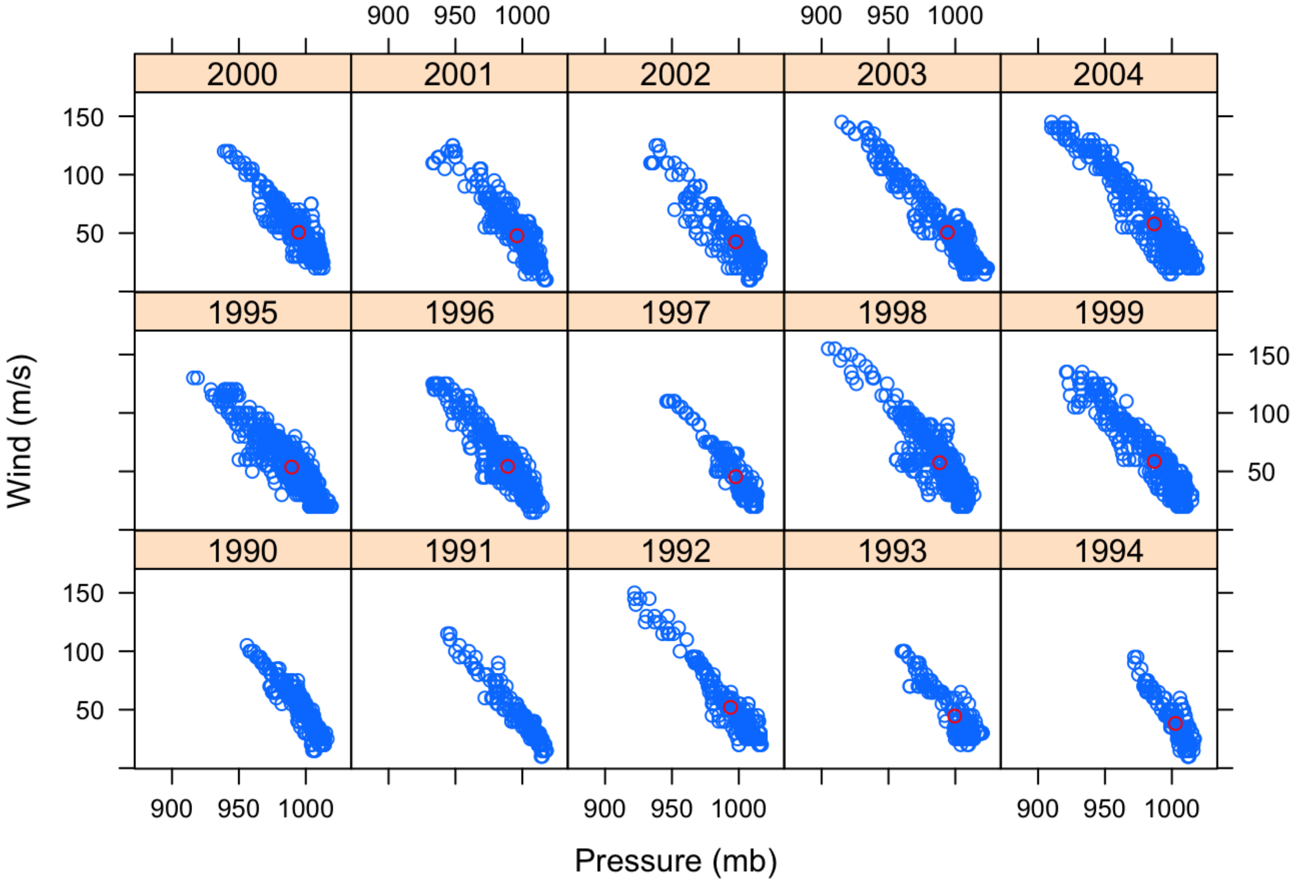


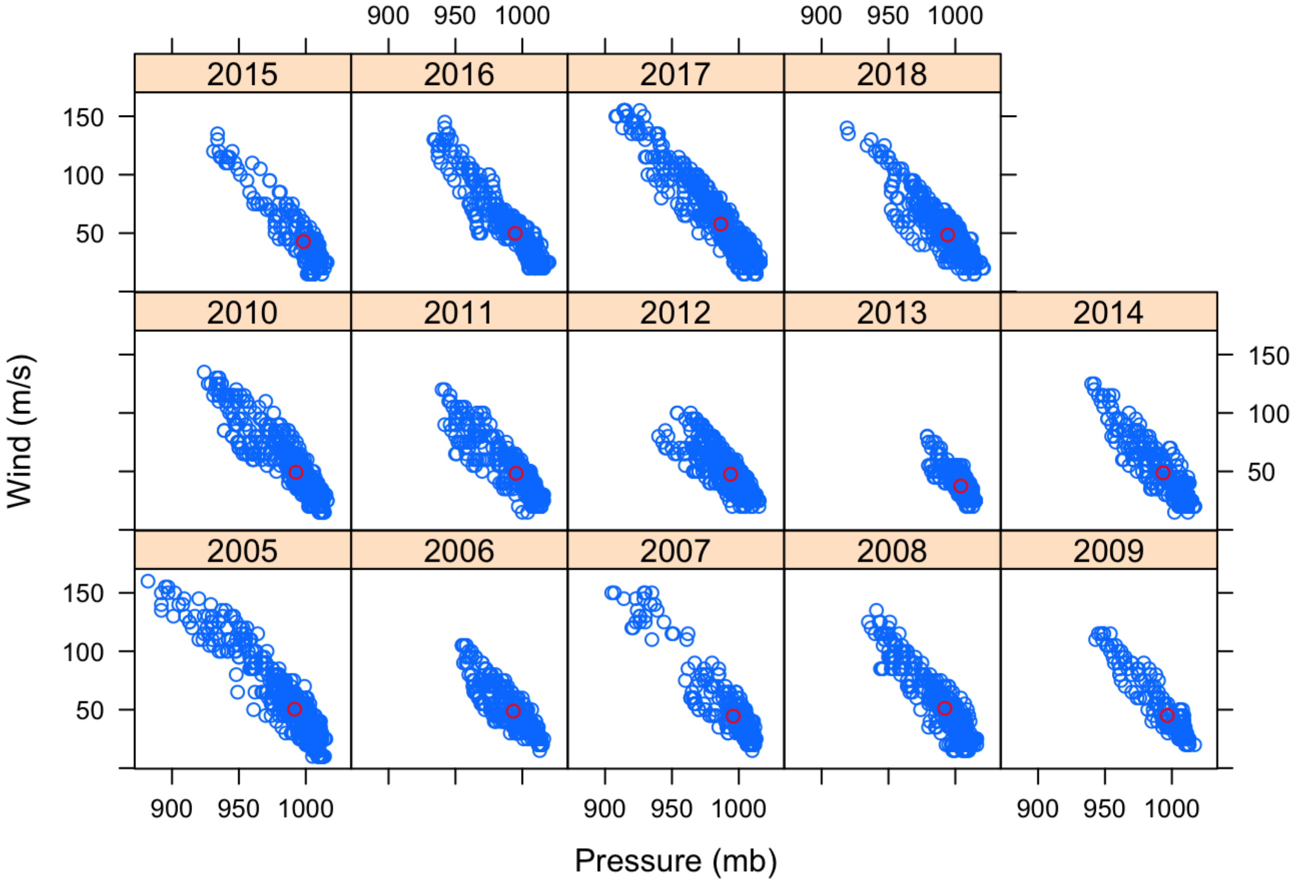


2.2.1 Panel functions

Example 1: Use `panel.points` to indicate mean values in each panel.

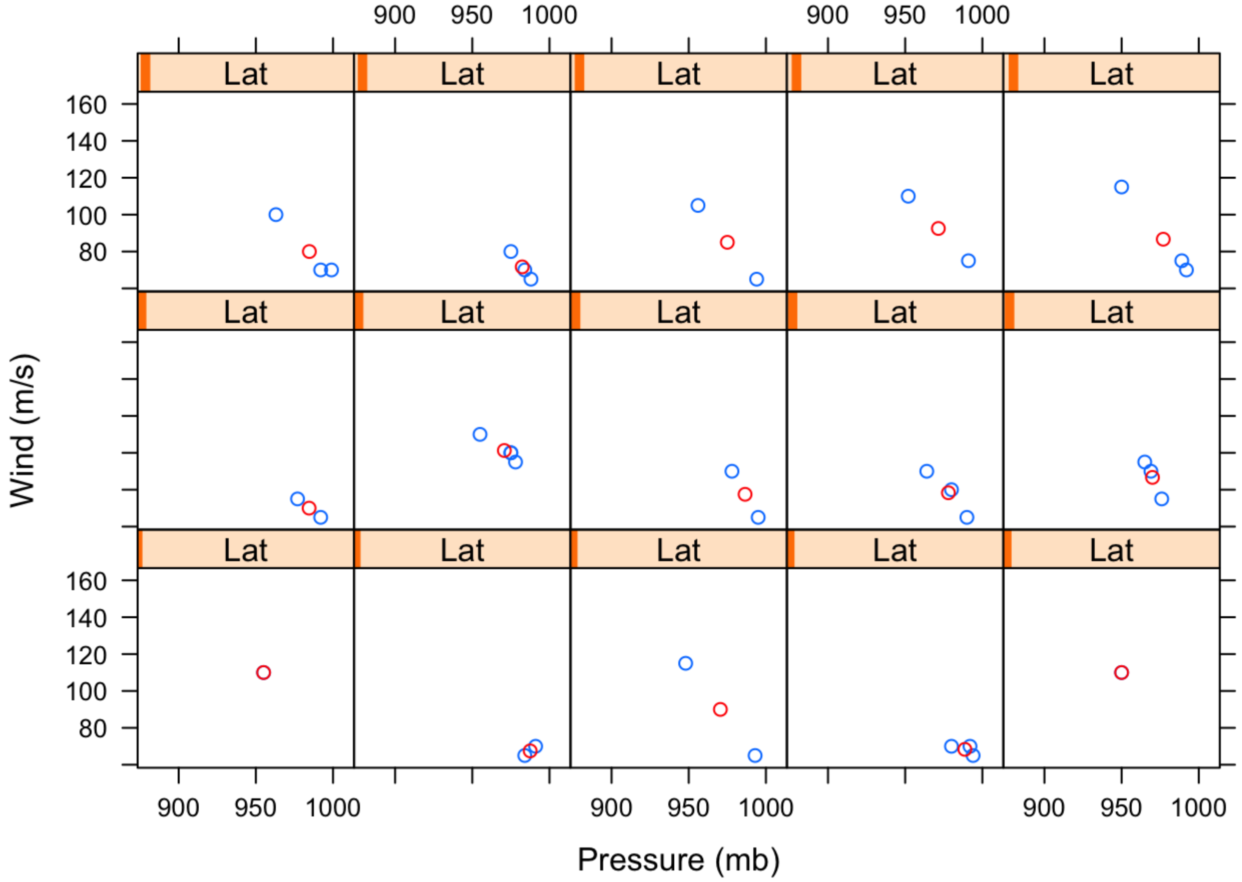
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       panel = function(x,y,...){
         panel.xyplot(x,y,...)
         panel.points(mean(x), mean(y), col="red")
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

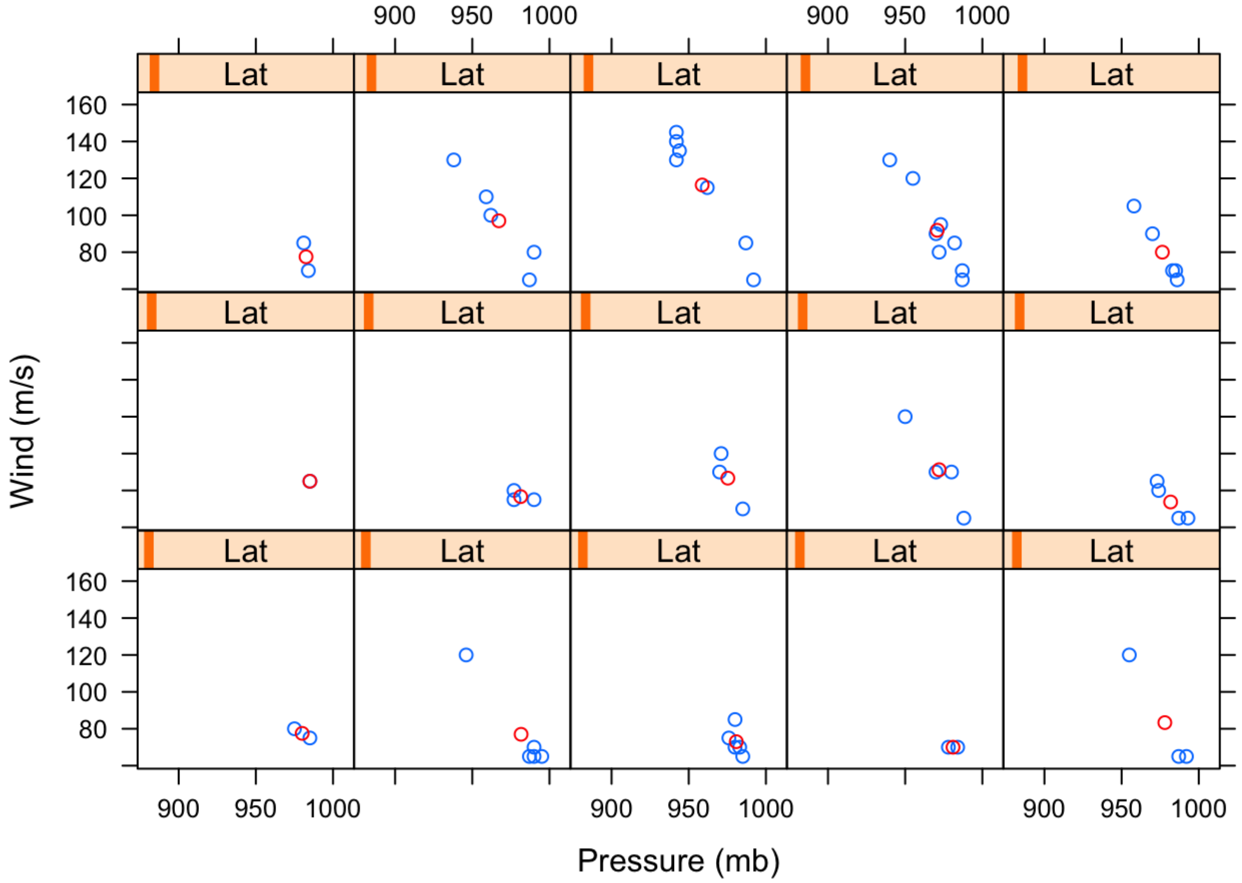


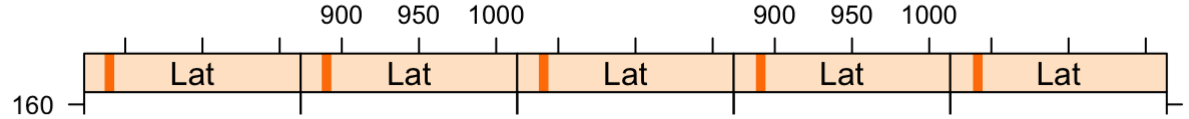


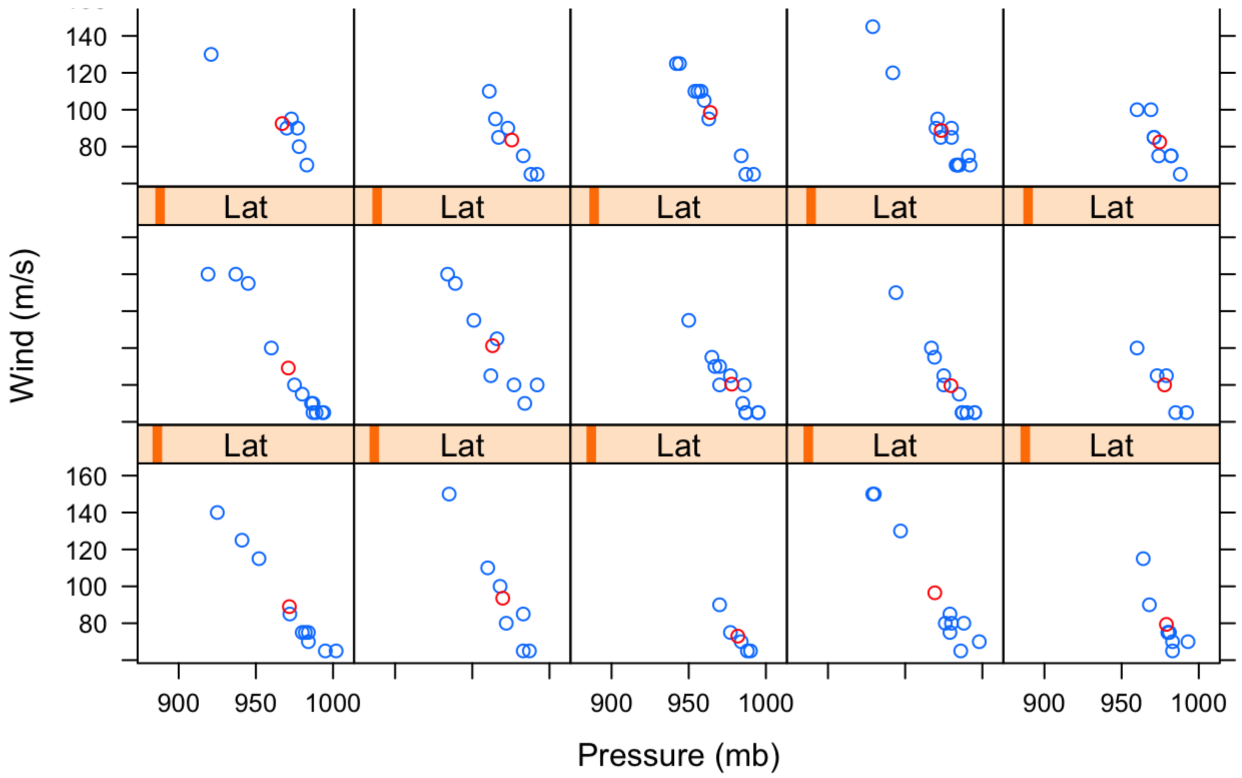
Example 2: Further conditioned on latitude (Lat). Note this generates many figures.

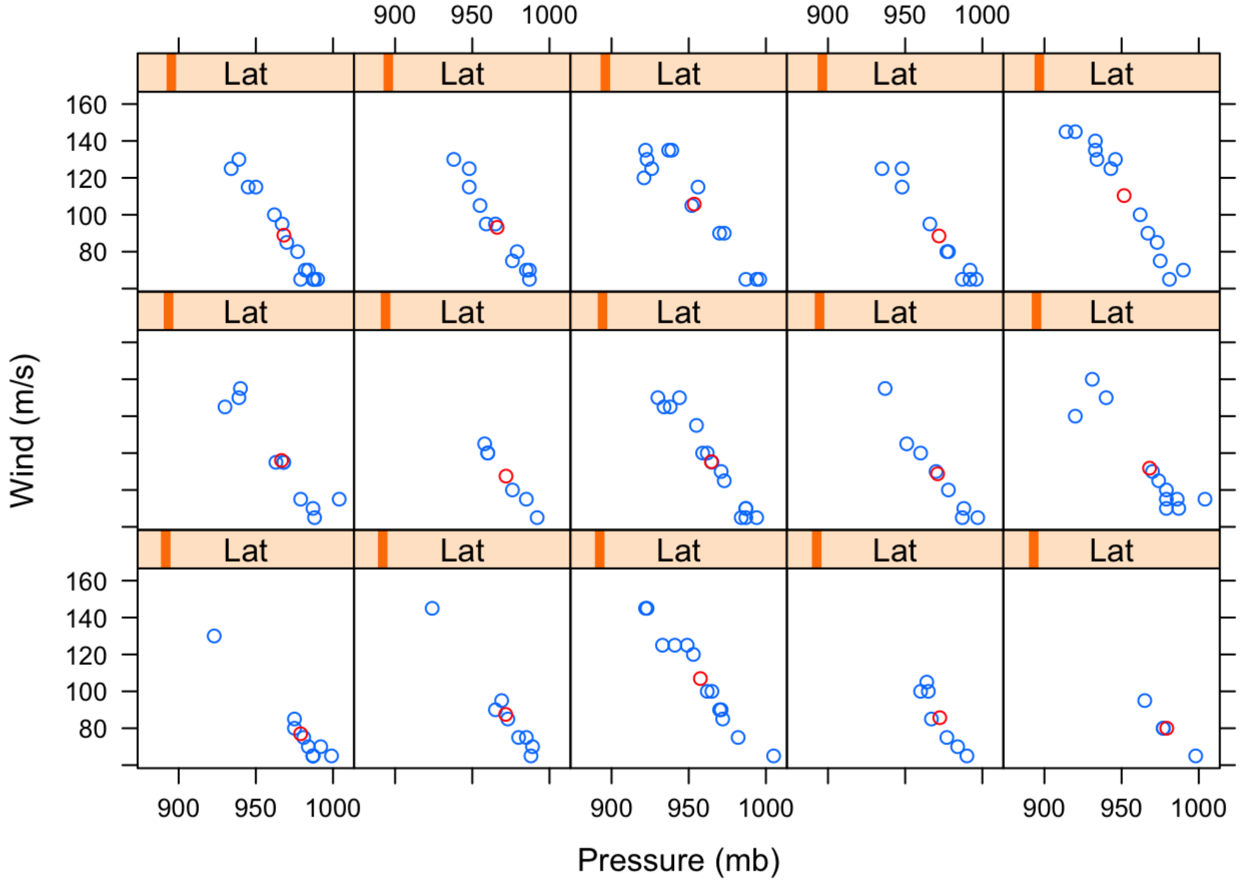
```
xyplot(Wind~Pressure | Lat,
       # try changing this to as.factor(Lat)
       data=subset(AL,Year >= 1990 & Status == "HU" & Lat >=10),
       layout=c(5,3),
       panel = function(x,y,...){
         panel.xyplot(x,y,...)
         panel.points(mean(x), mean(y), col="red")
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

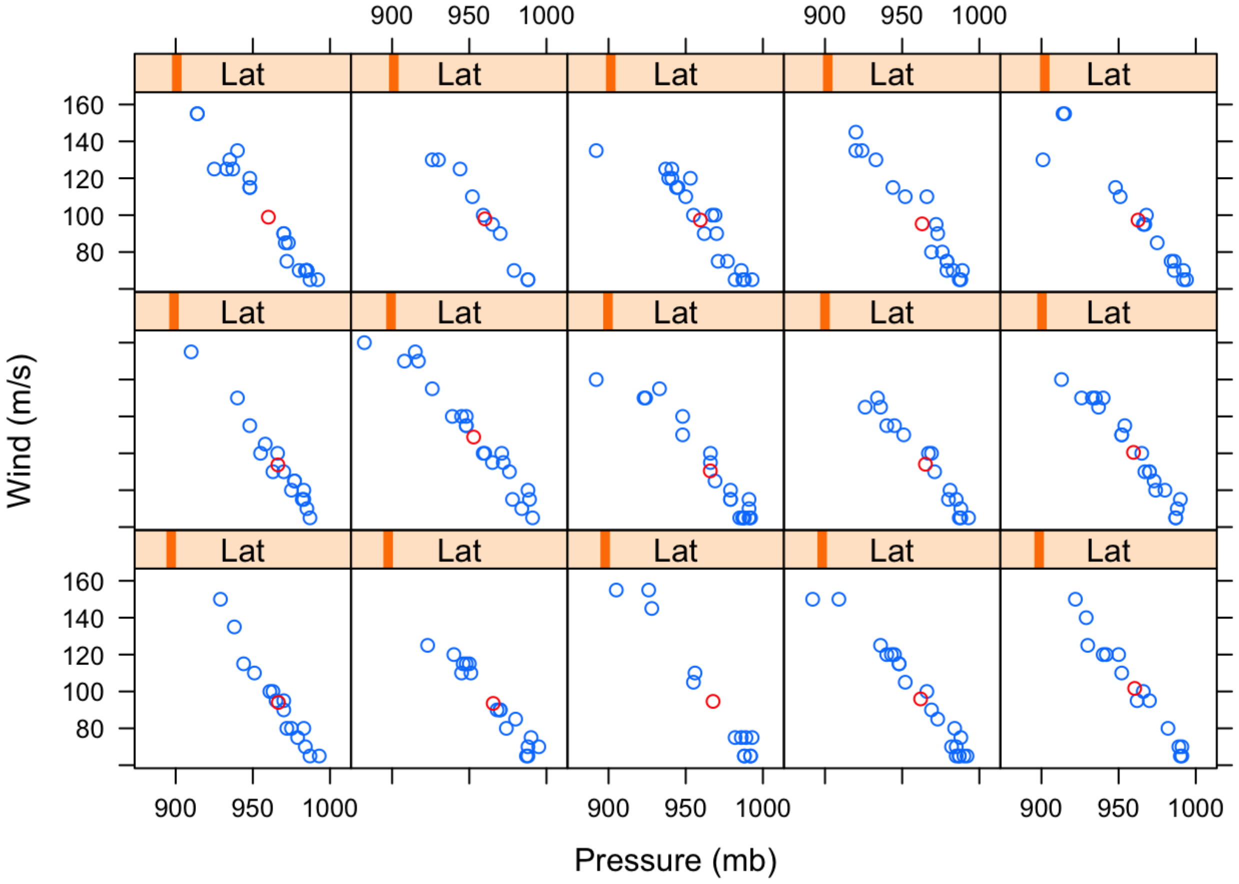

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

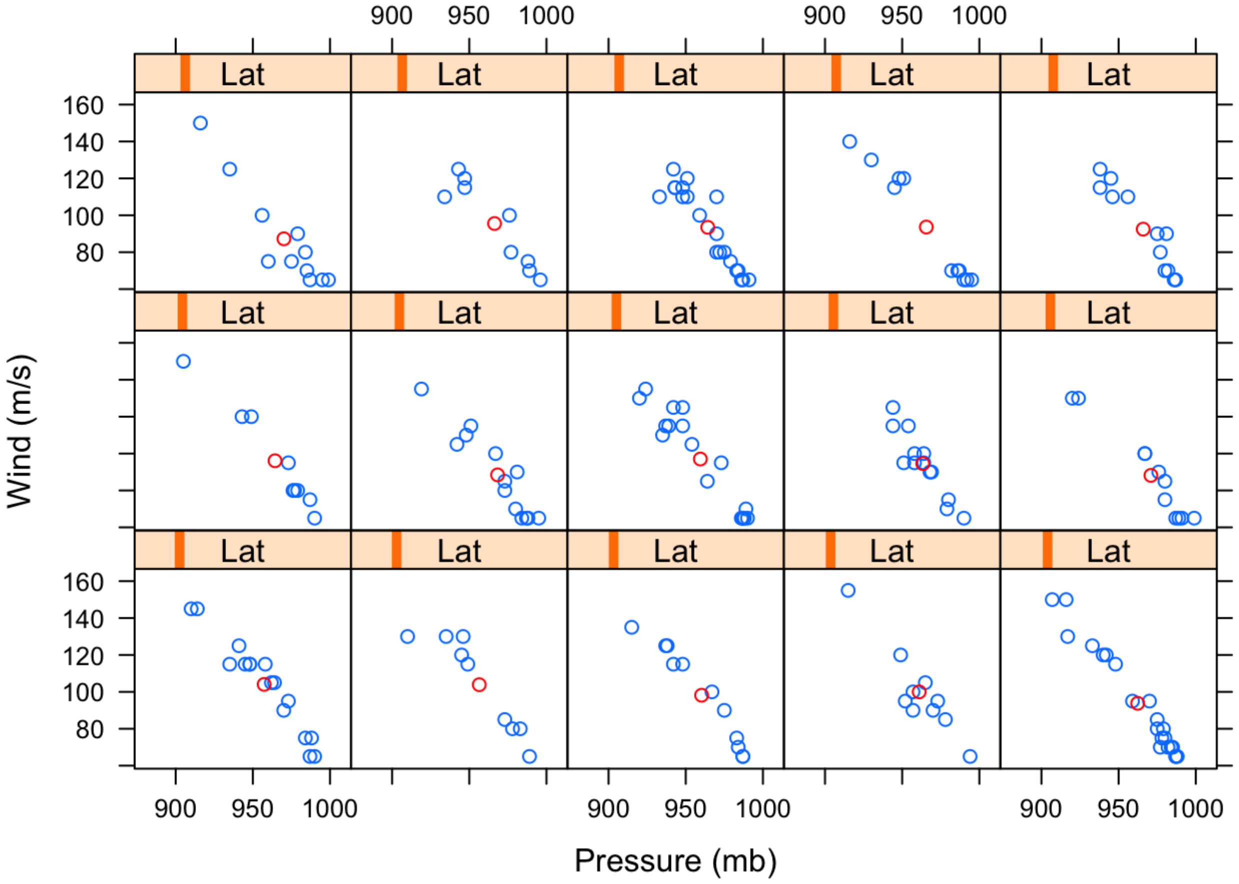

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

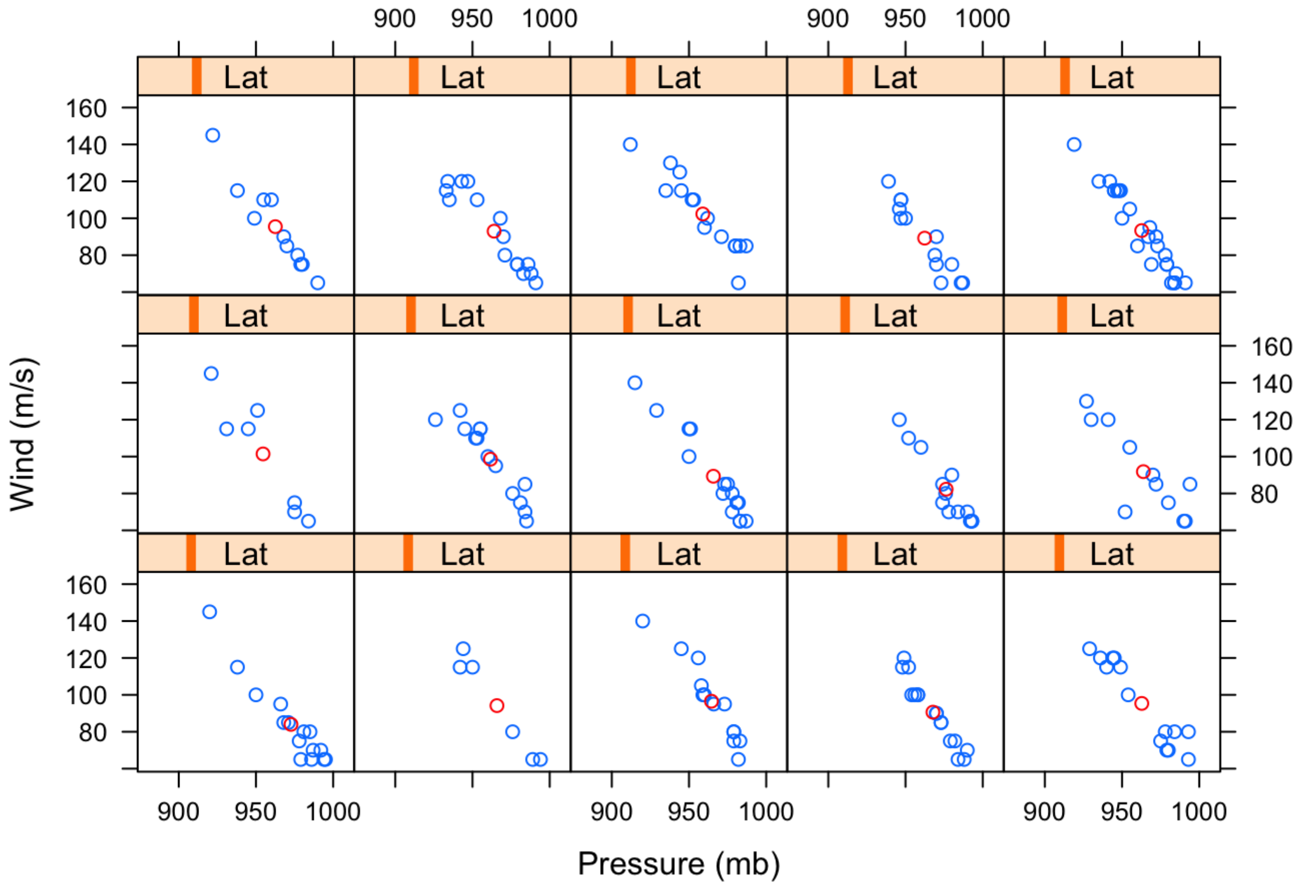

Lat
900
950
1000
160

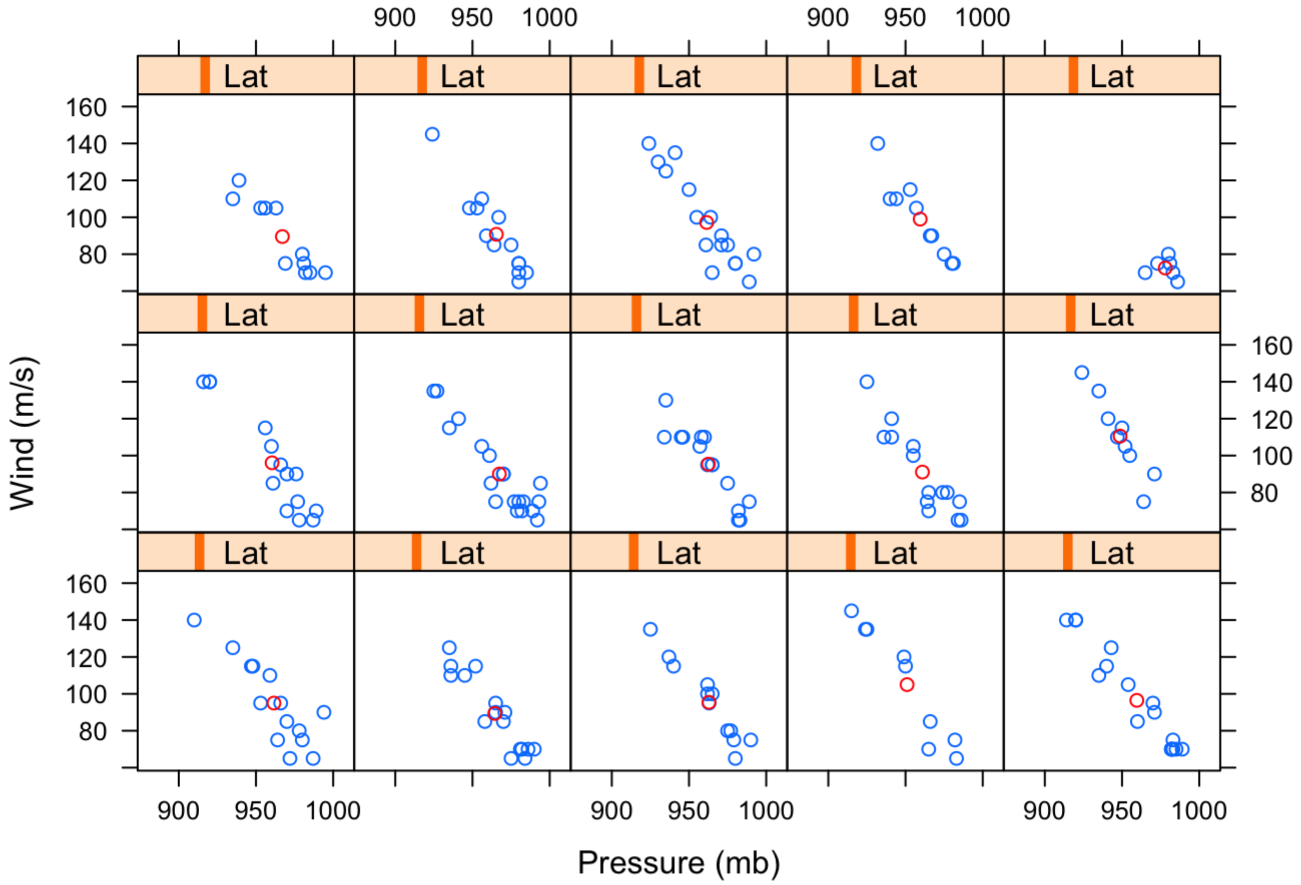

Wind (m/s)
Pressure (mb)
Lat
140
120
100
80
160
900
950
1000

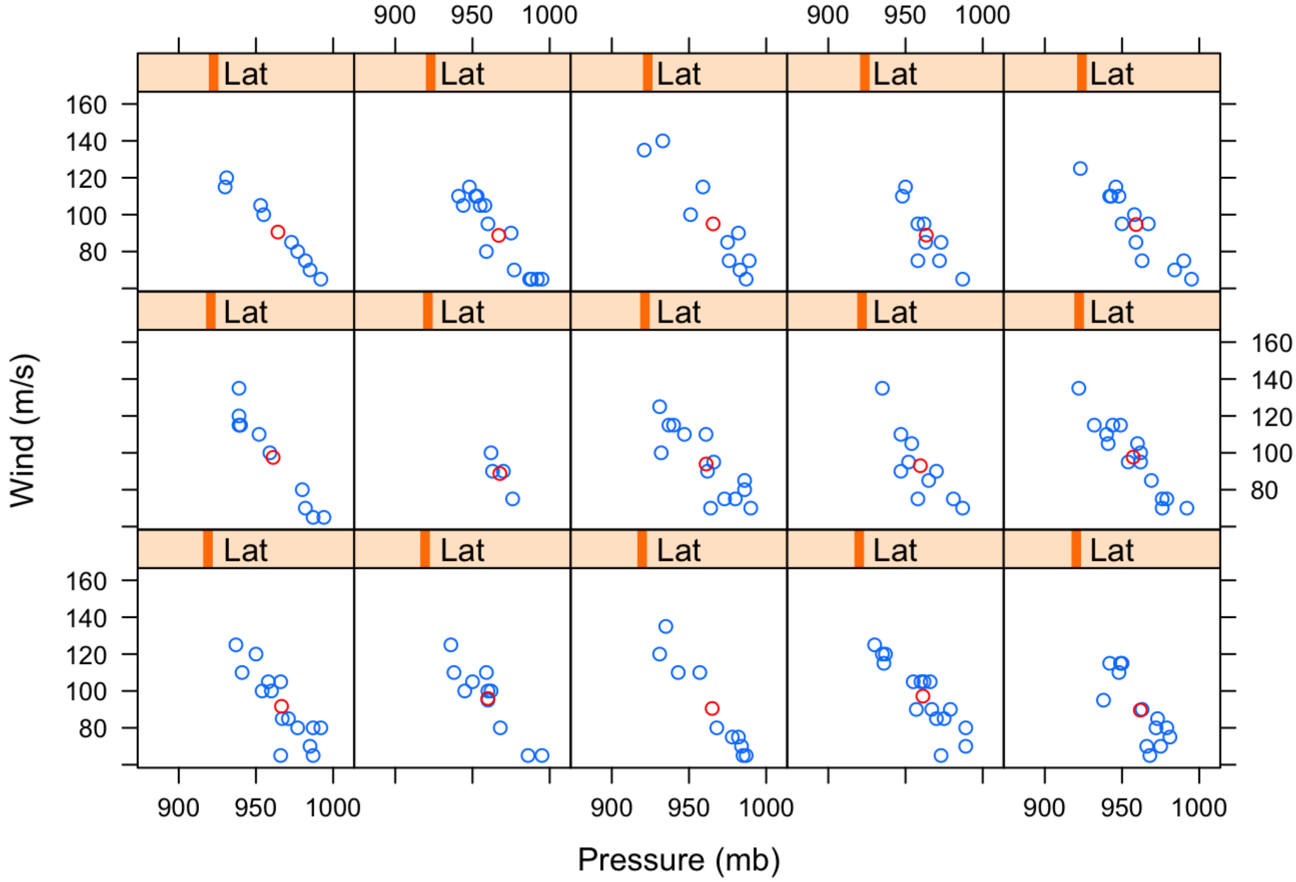

Wind (m/s)
Pressure (mb)
Lat
140
120
100
80
160
900
950
1000

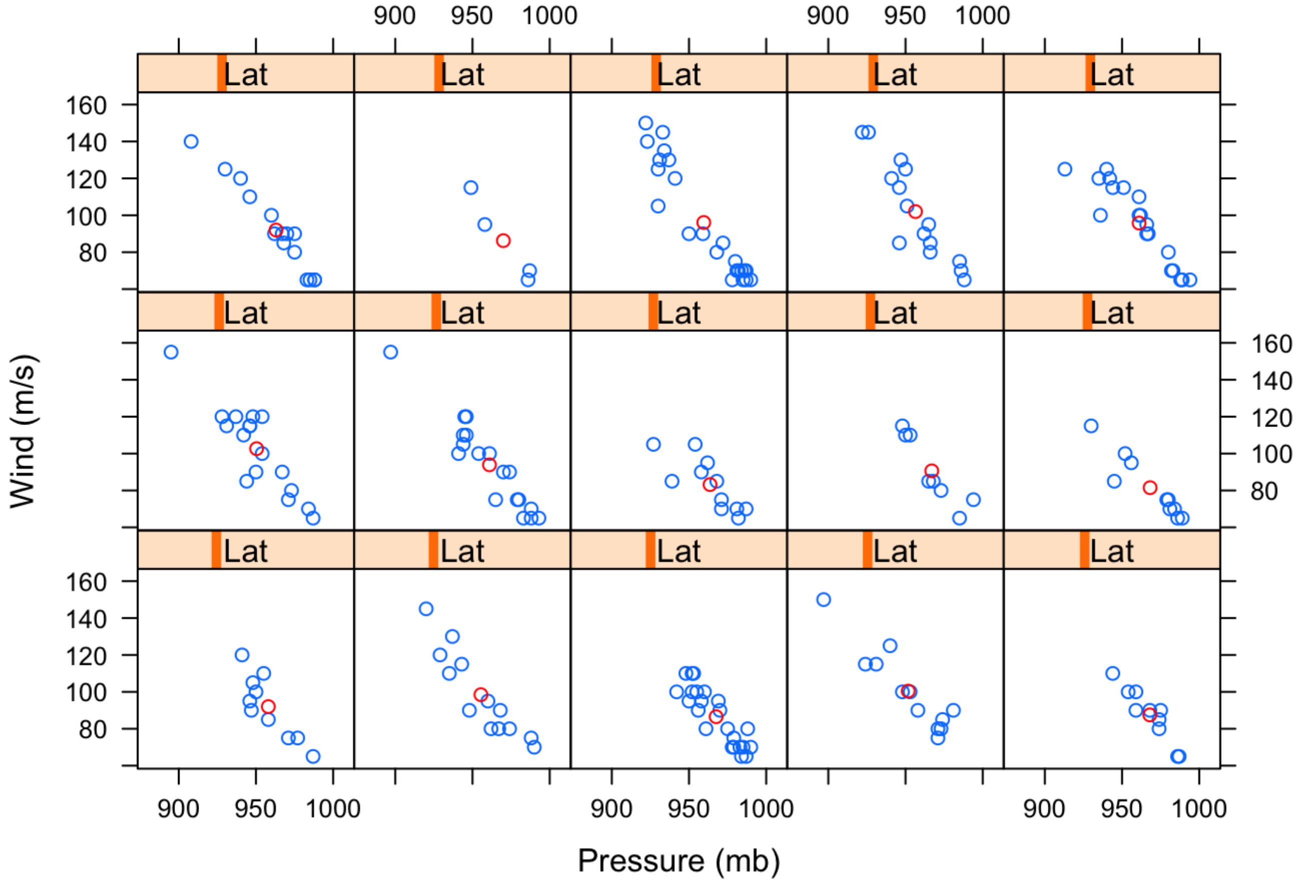

Lat
Wind (m/s)
Pressure (mb)
160
140
120
100
80
900
950
1000

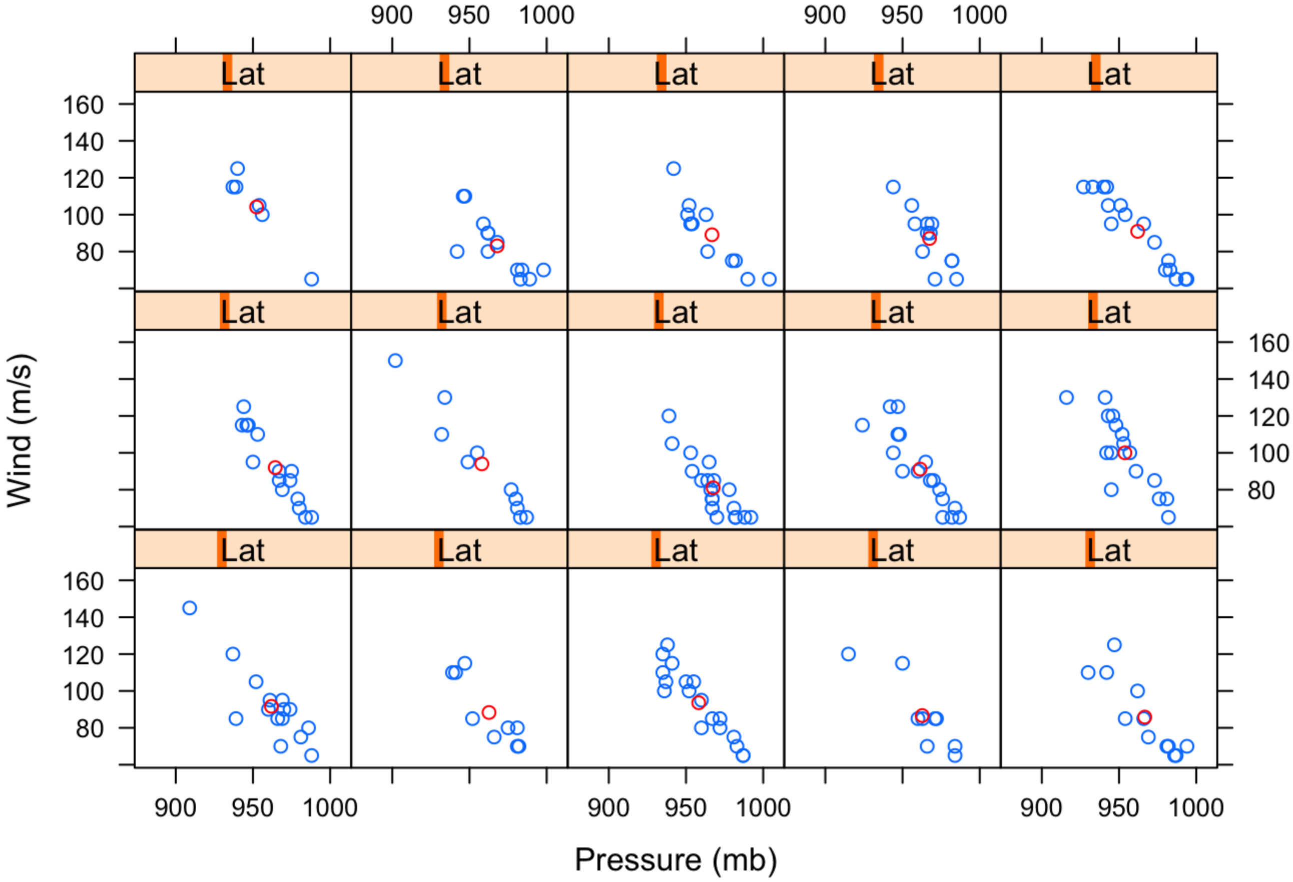

Lat
Wind (m/s)
Pressure (mb)
160
140
120
100
80
900
950
1000

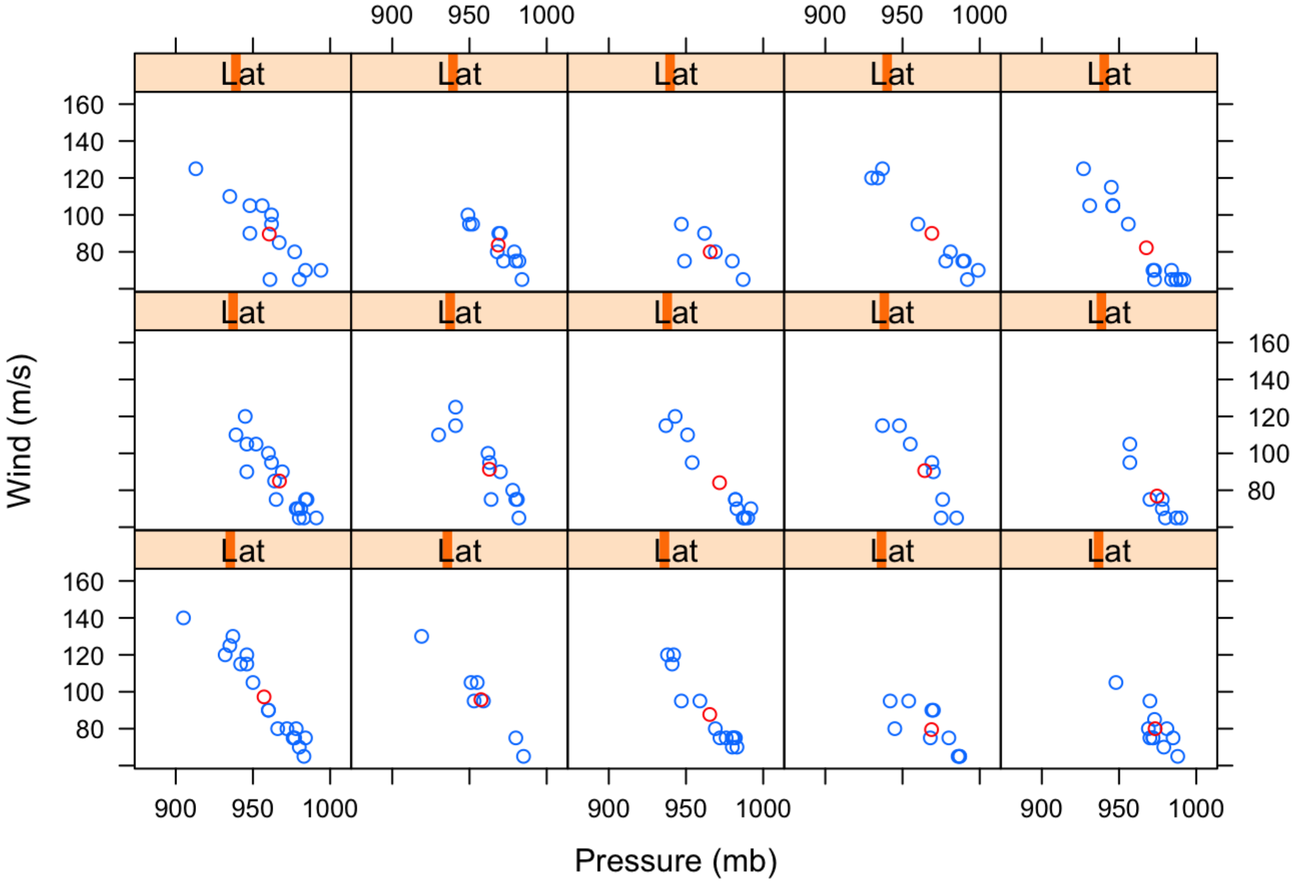

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

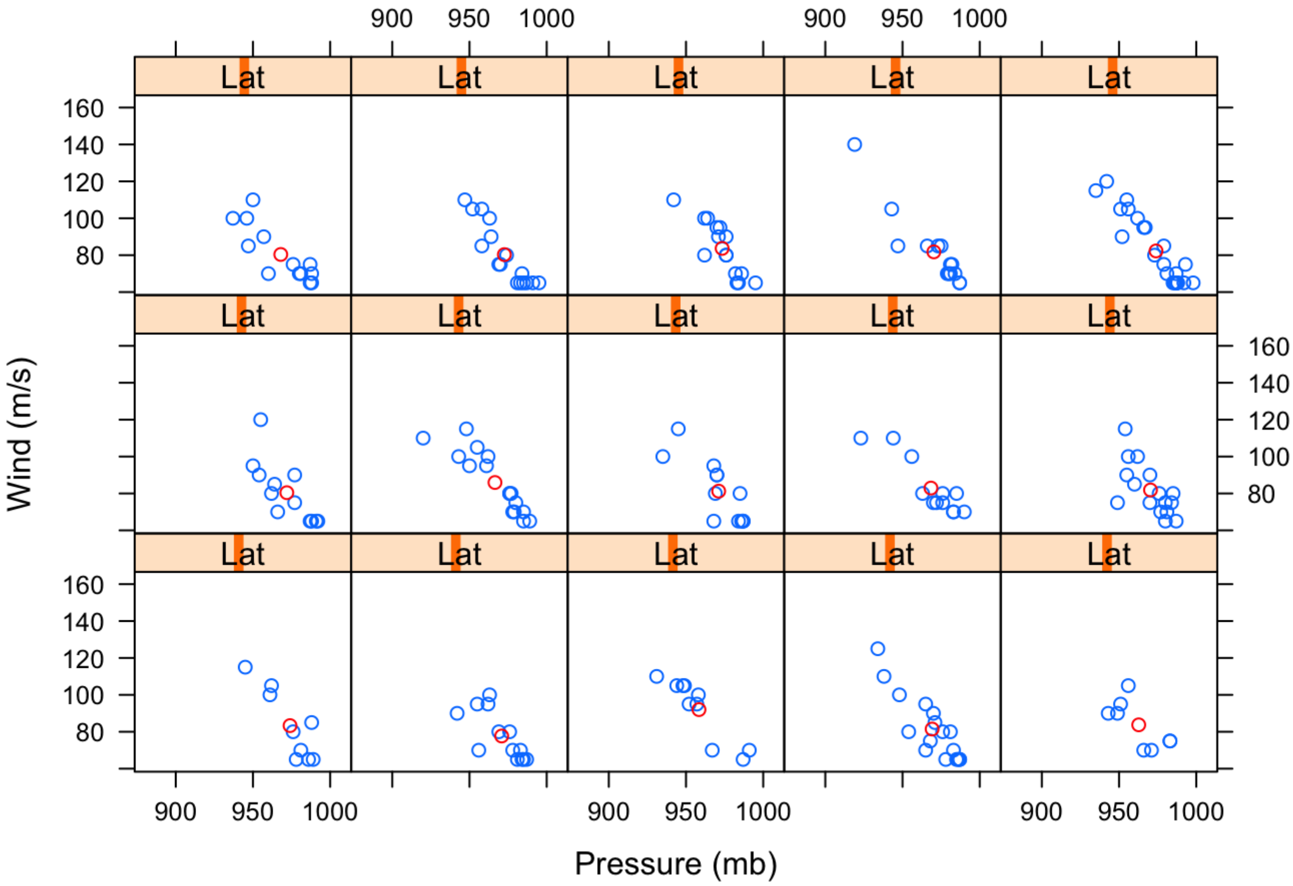

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

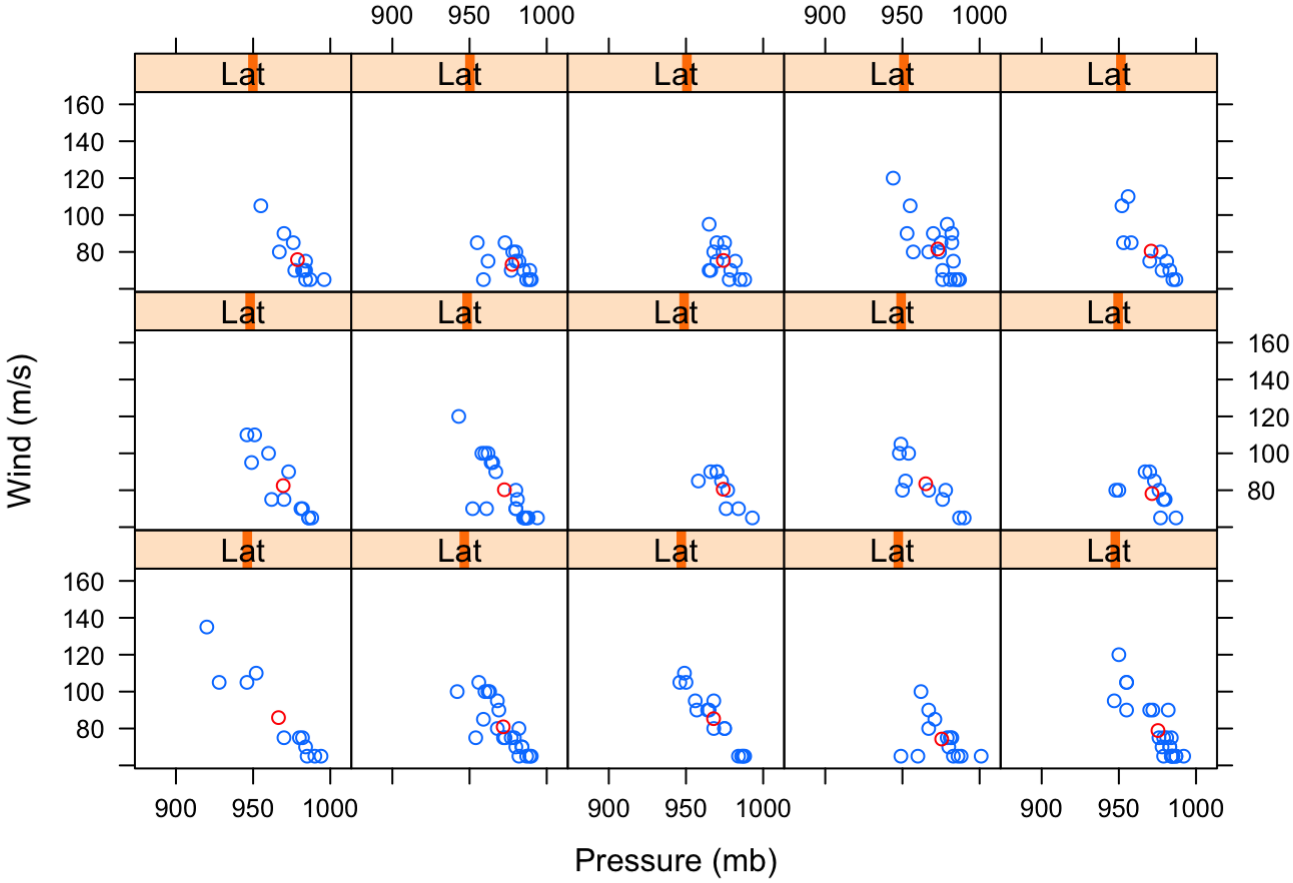

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

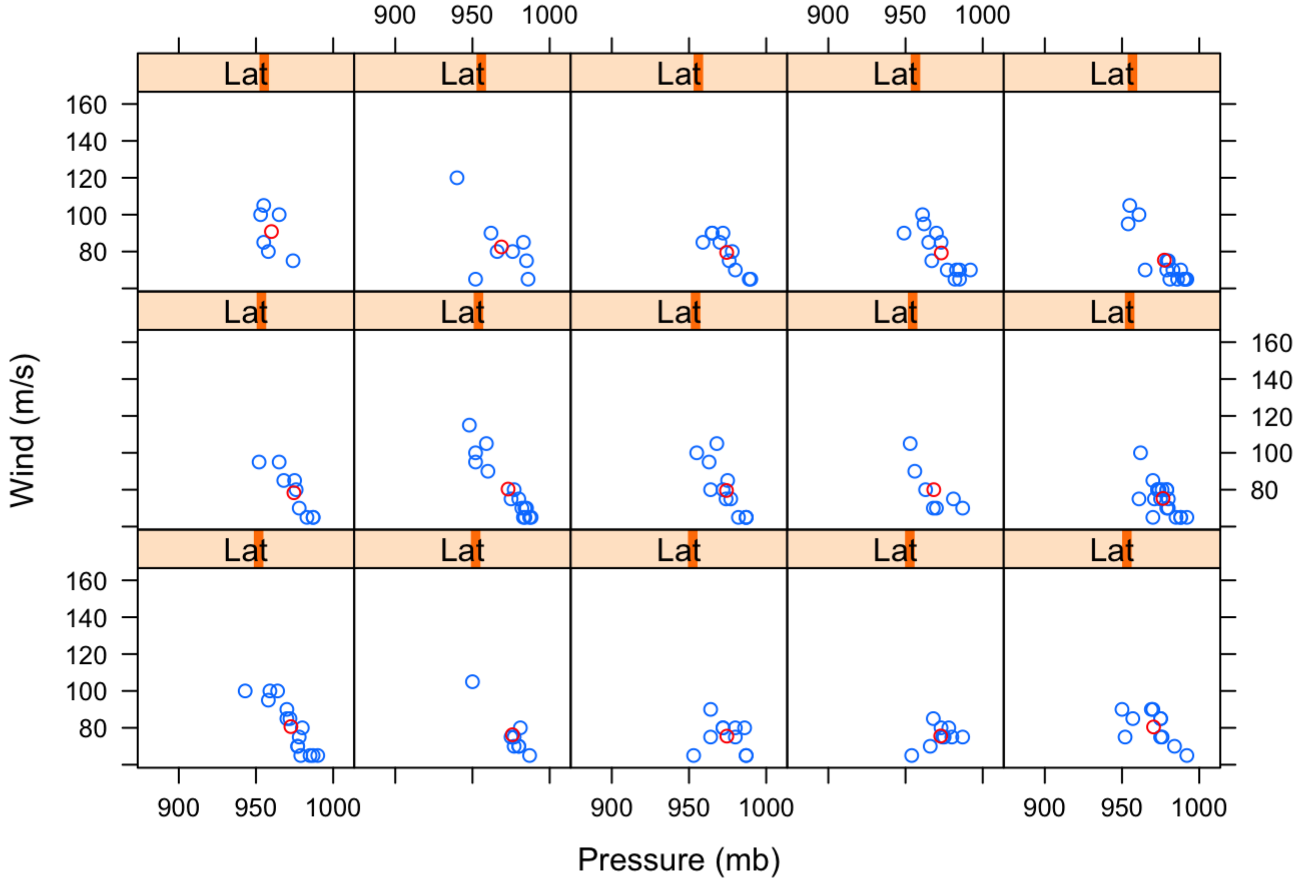

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

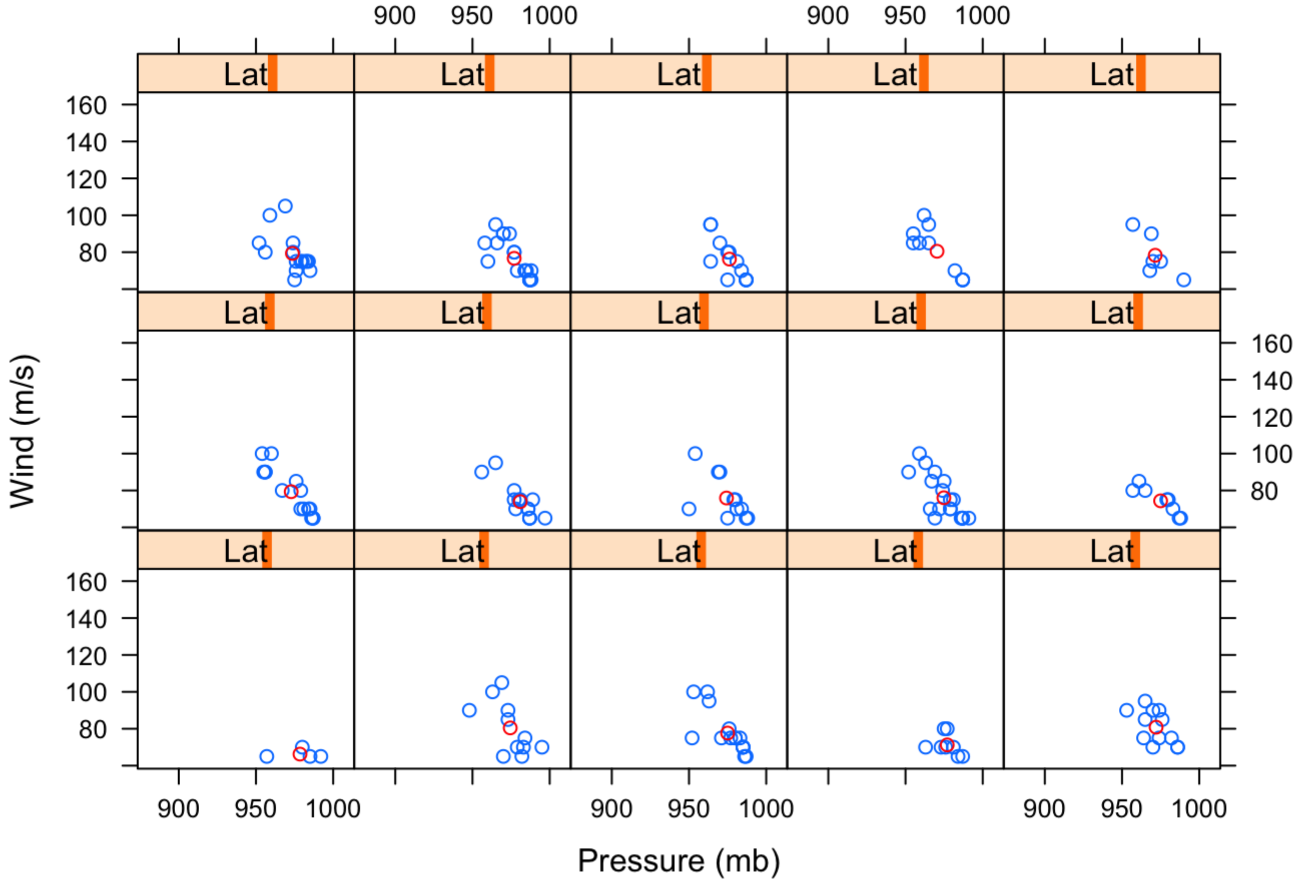

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

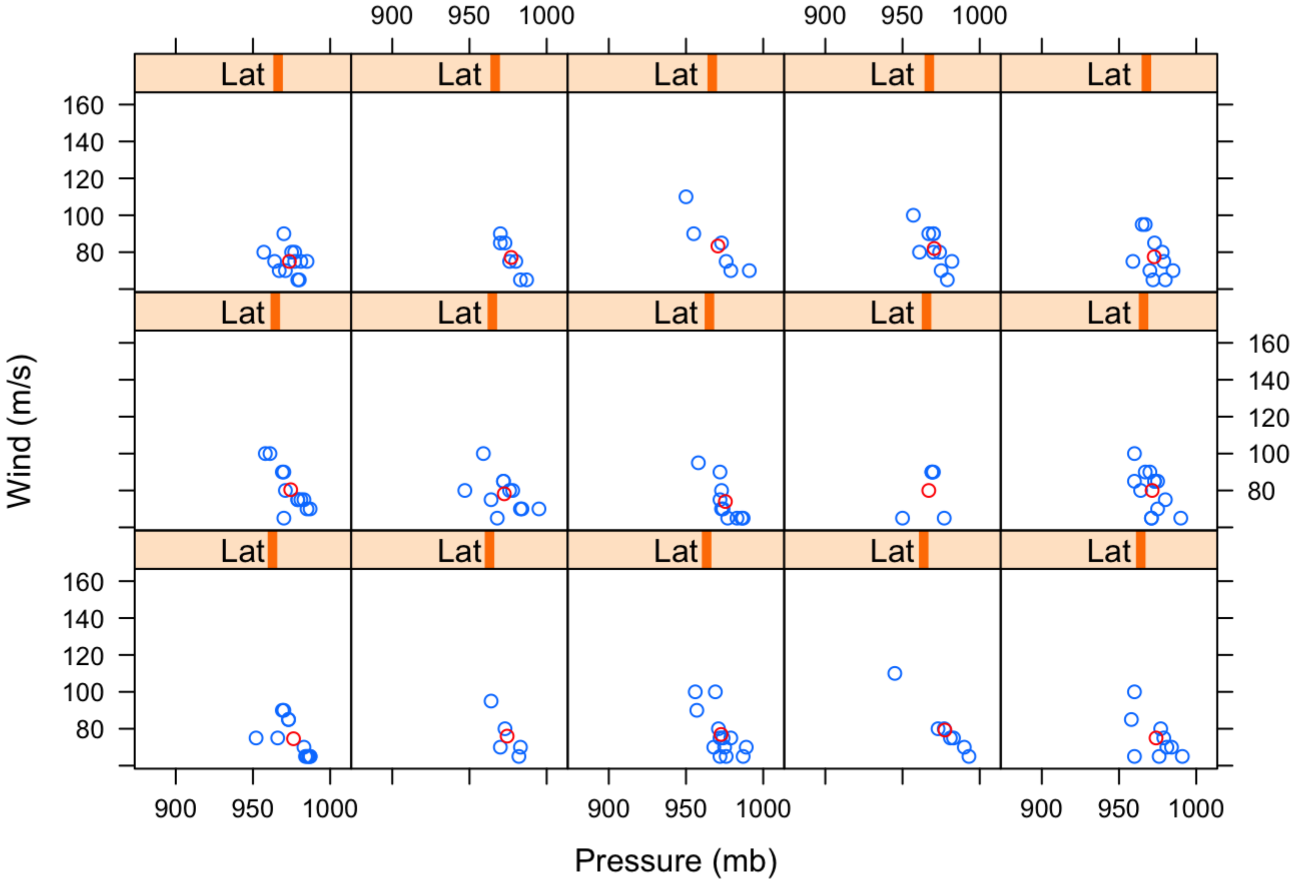

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

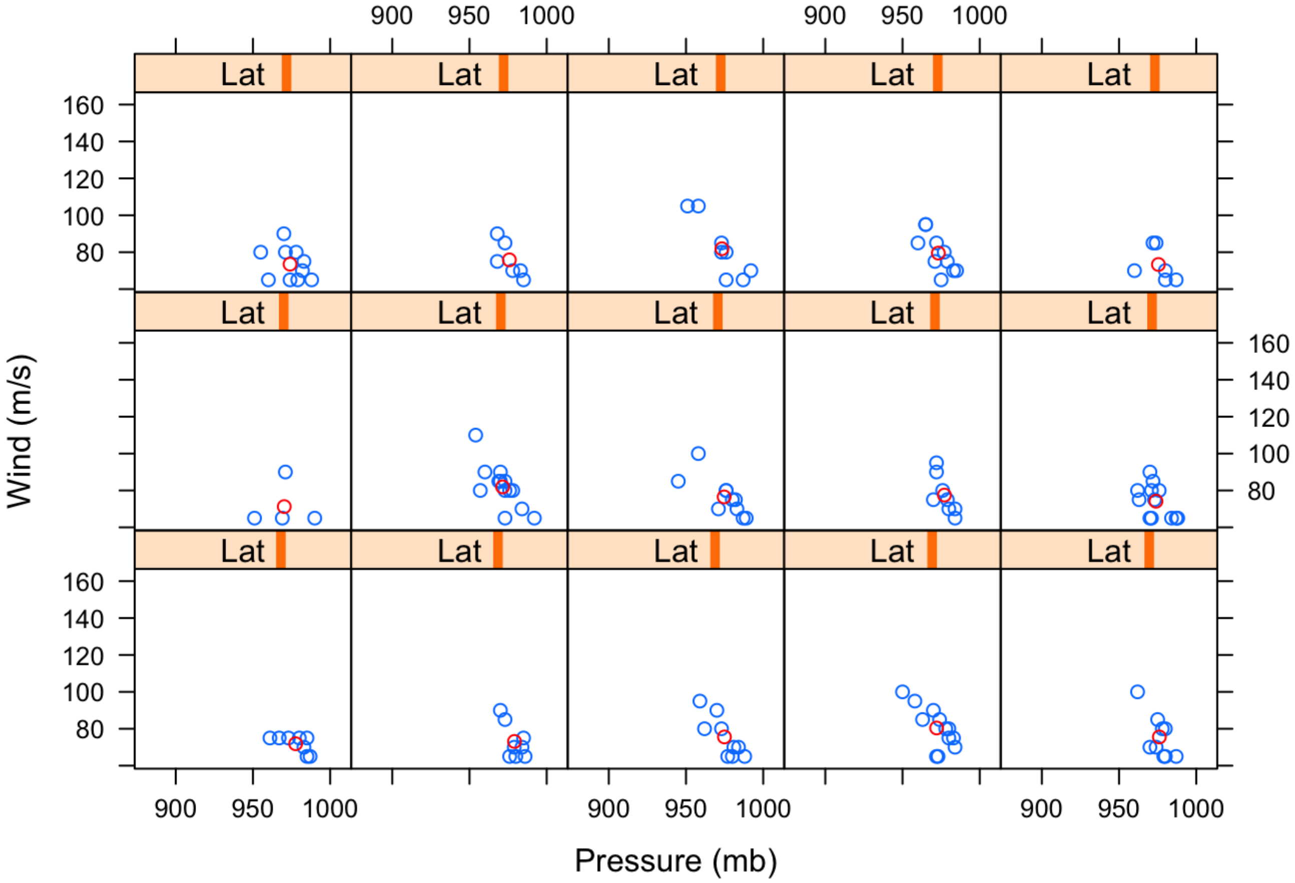

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

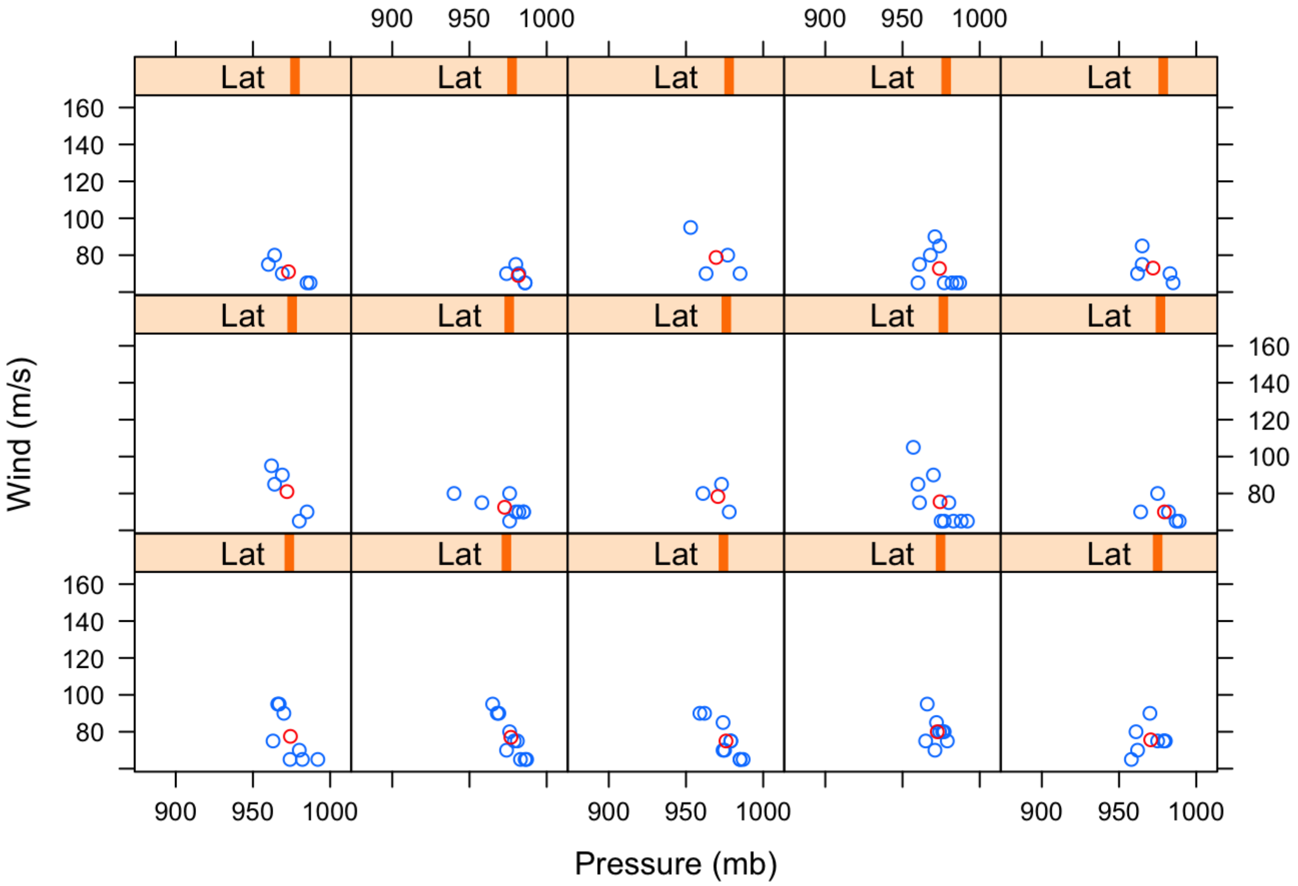

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

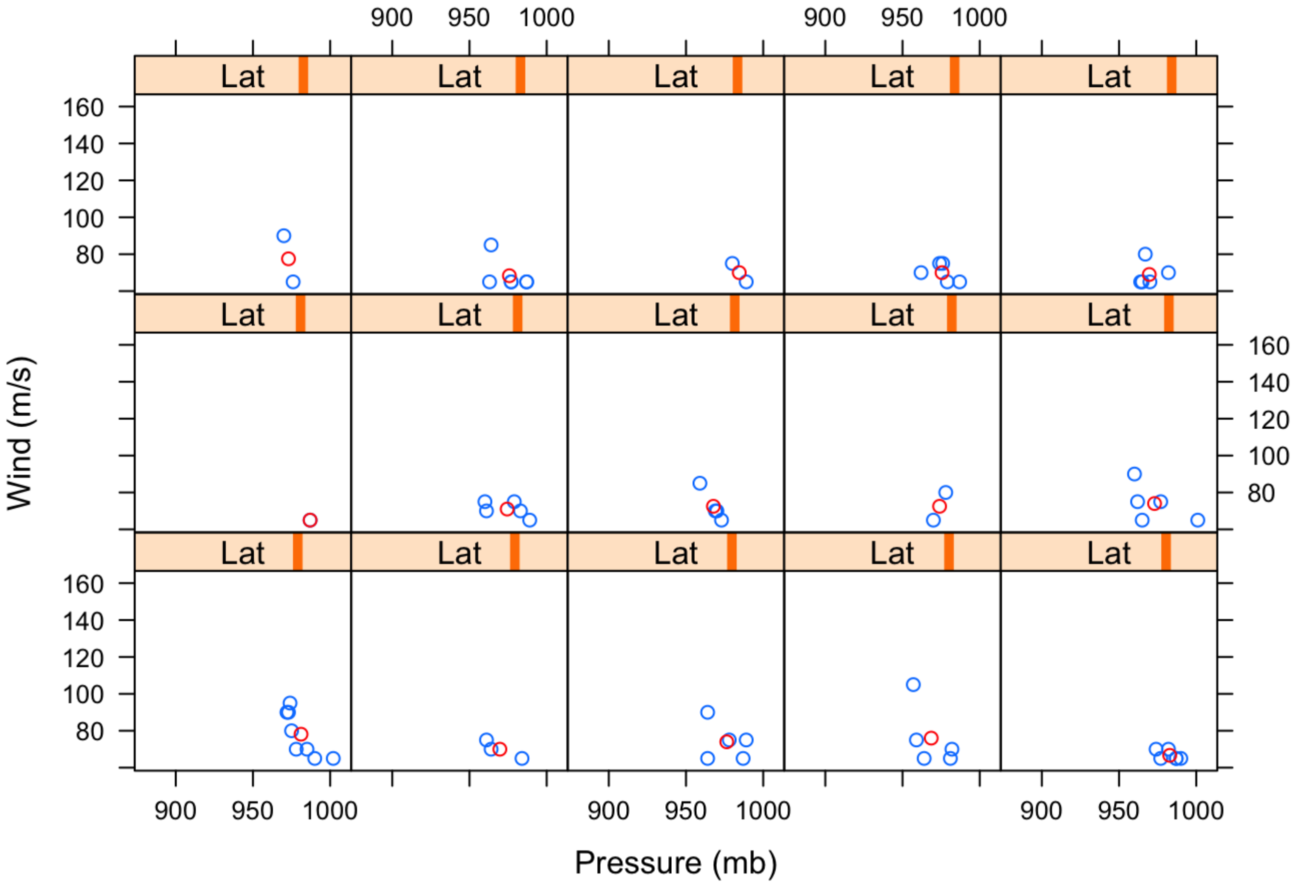

Lat
900
950
1000
160
140
120
100
80
Wind (m/s)
Pressure (mb)

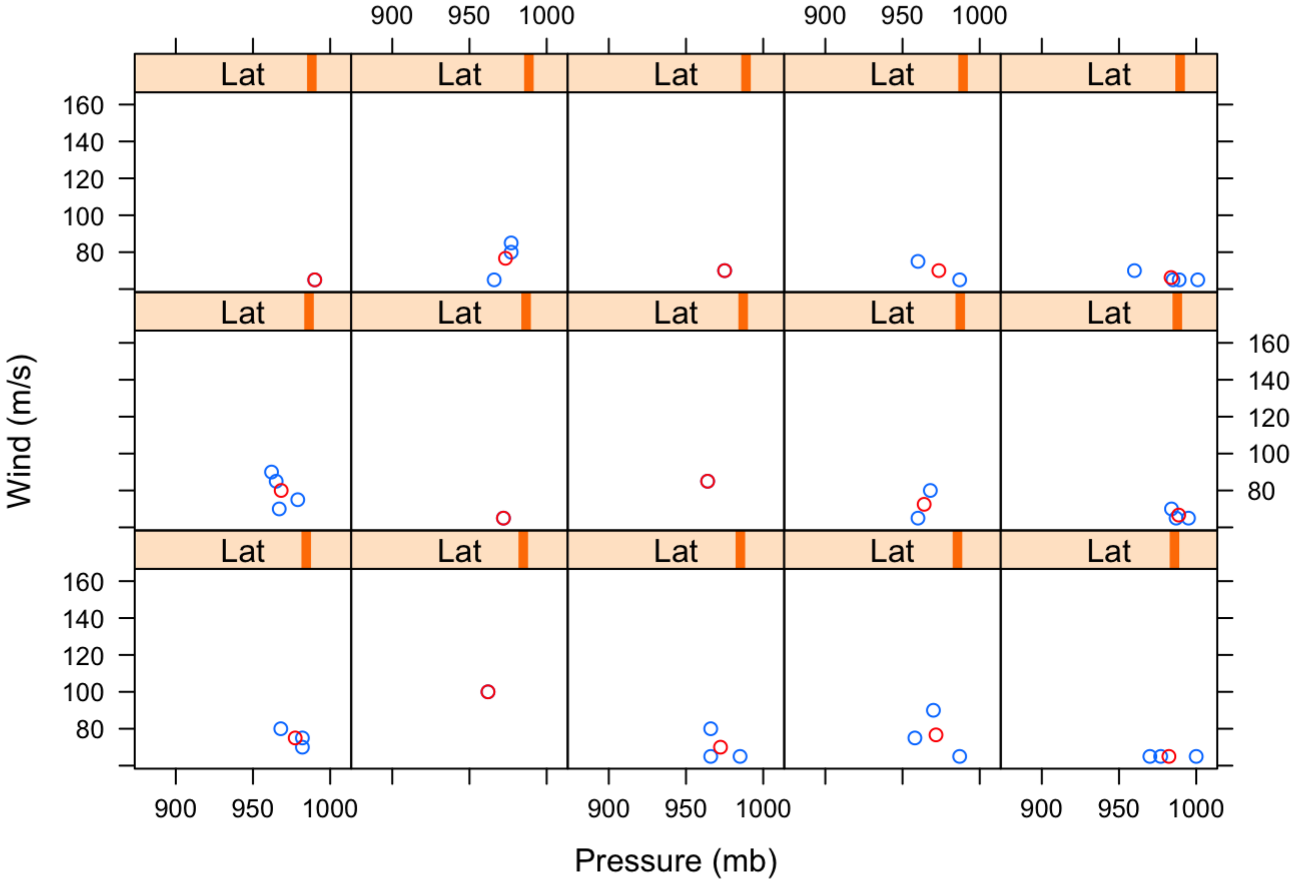

Lat
900
950
1000
160
140
120
100
80
Wind (m/s)
Pressure (mb)

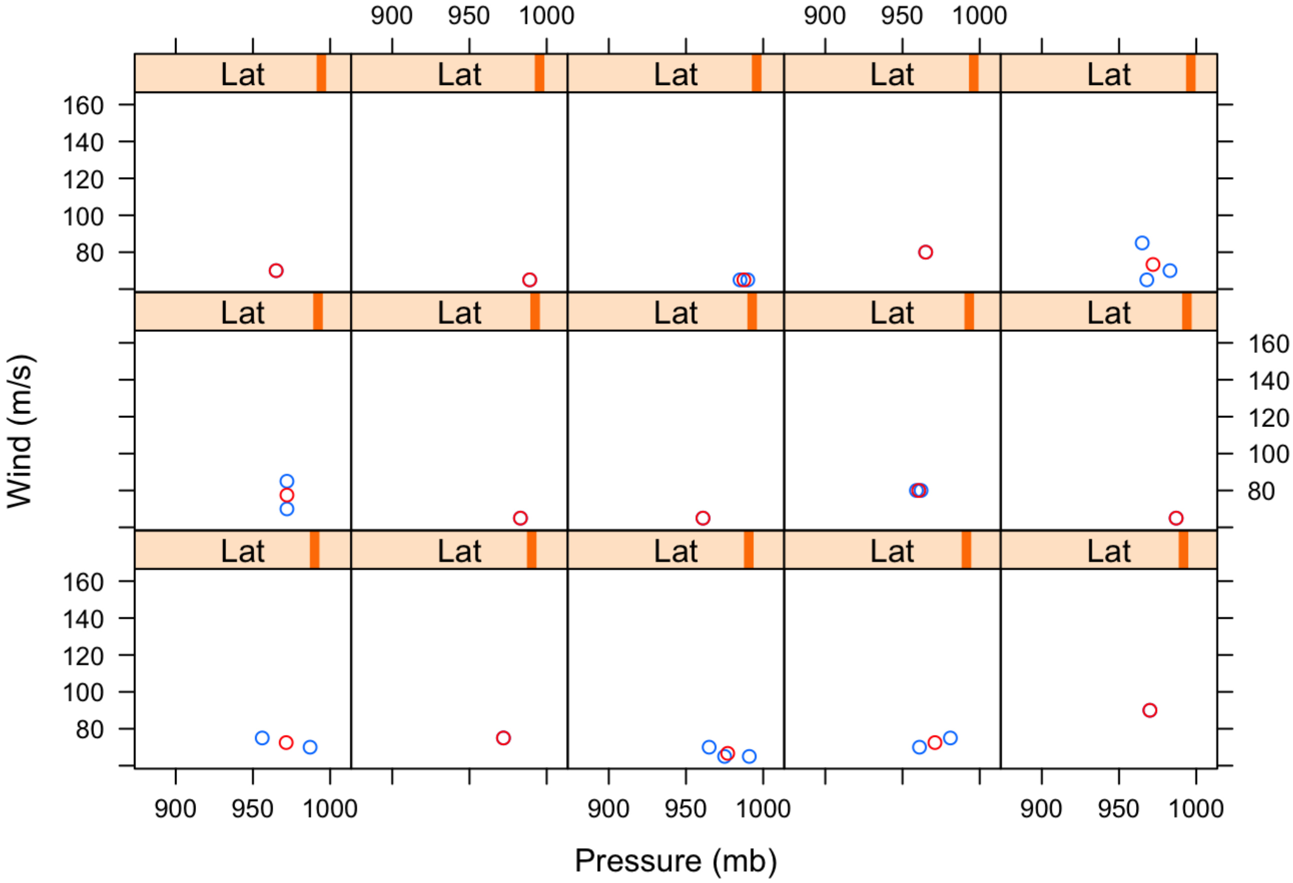

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

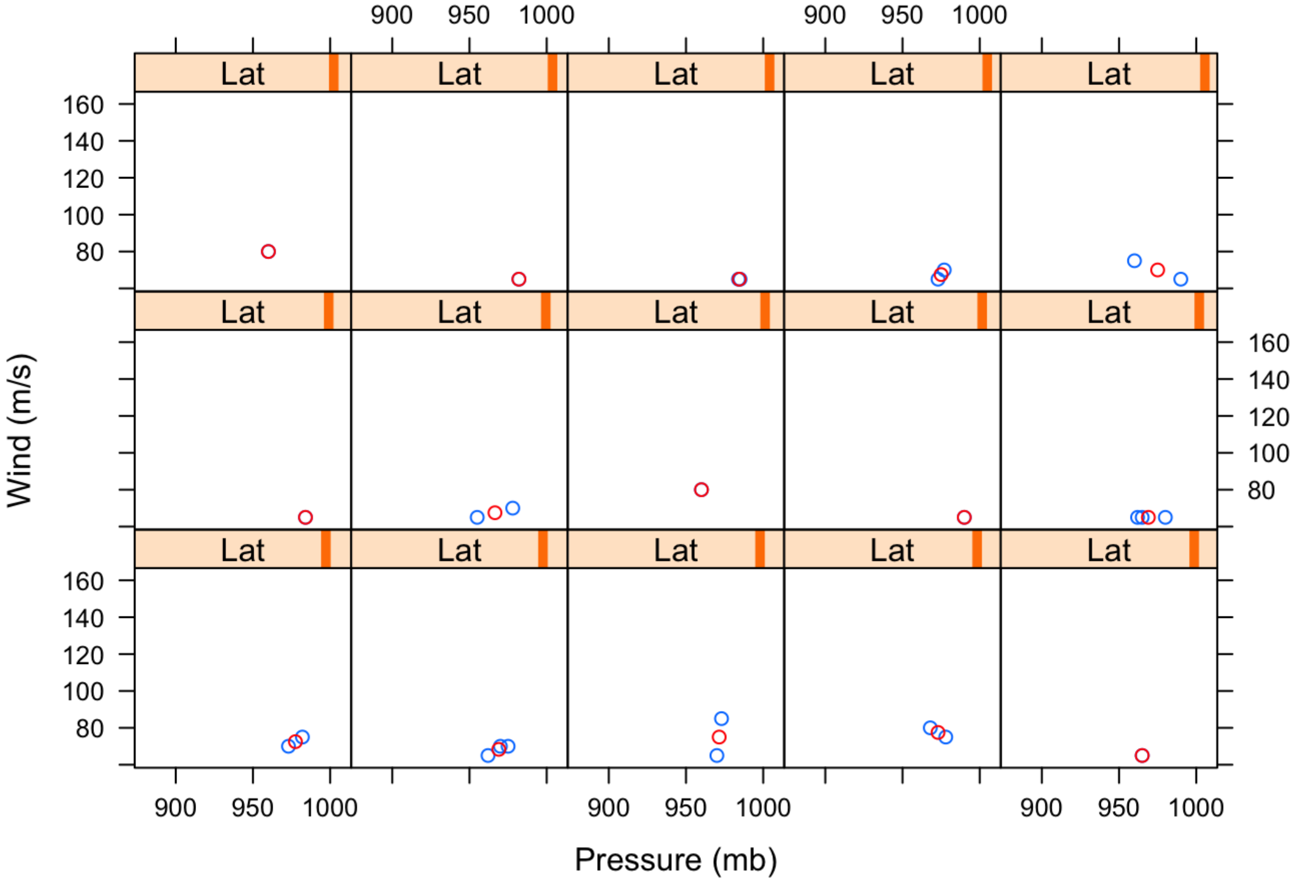

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

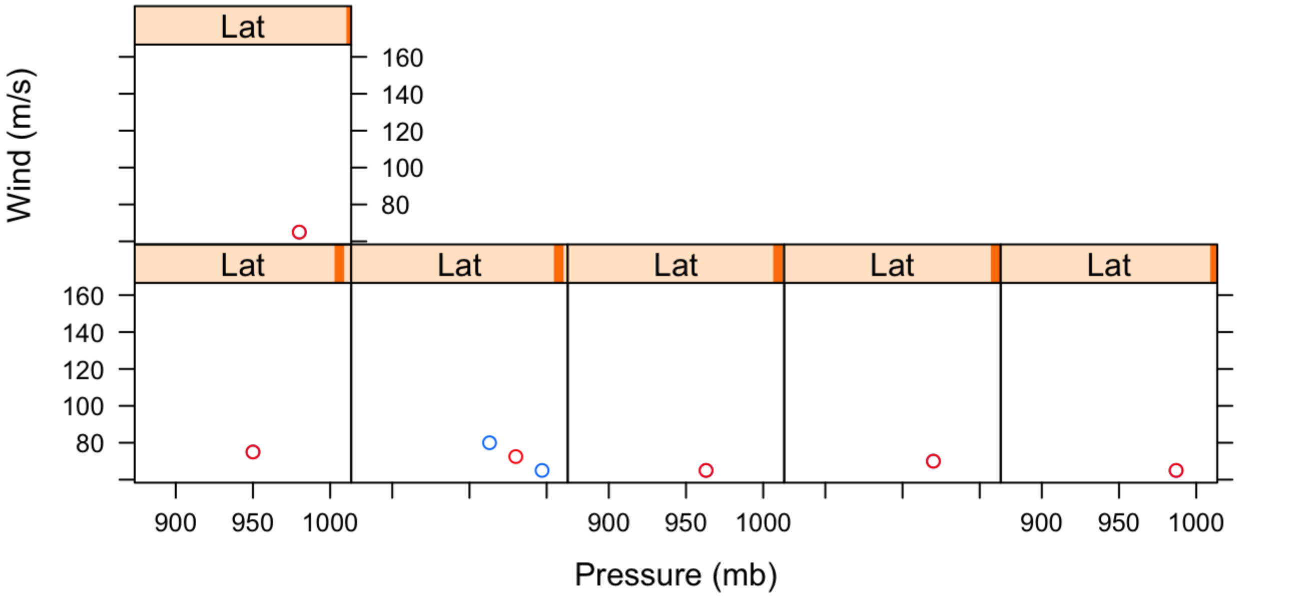

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160


Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160


Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

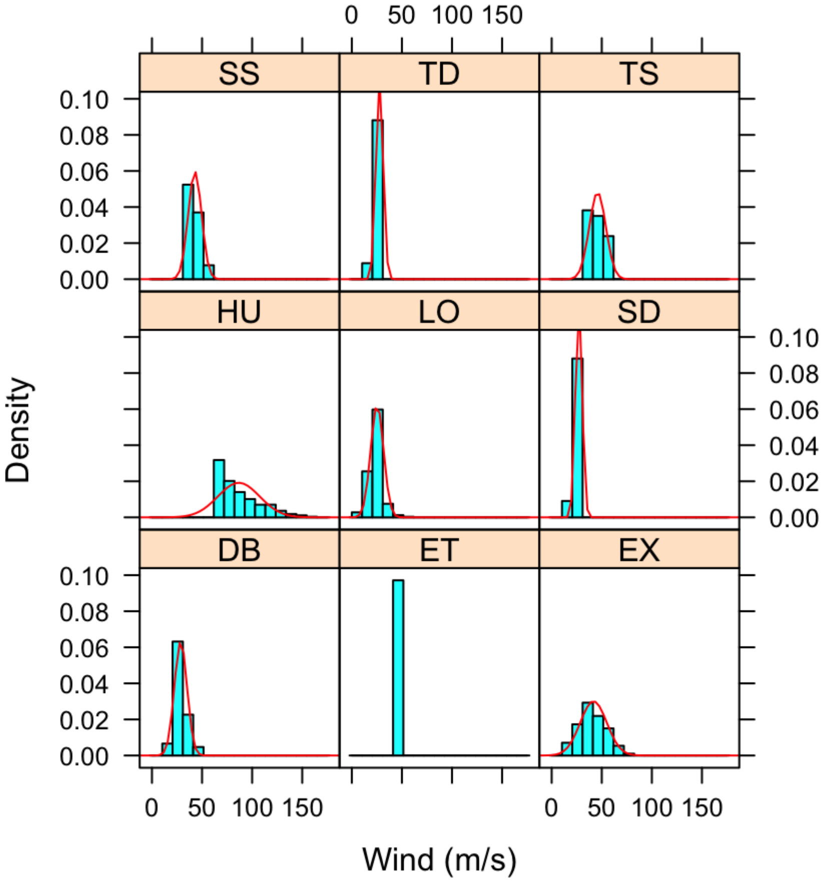

Lat
Wind (m/s)
Pressure (mb)
900
950
1000
80
100
120
140
160

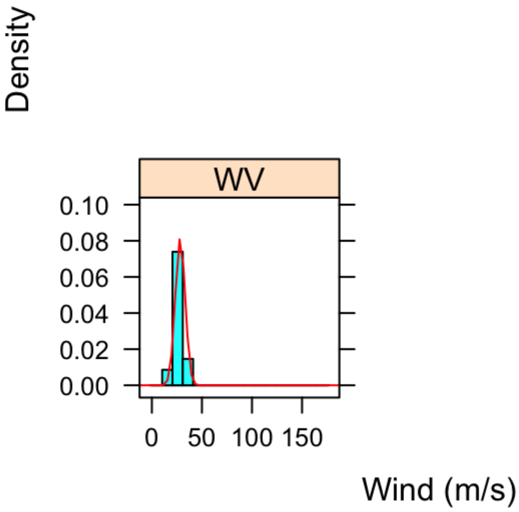




Example 3: Use `panel.mathdensity` to estimate and plot a theoretical probability density function (normal in this case) given data in each panel.

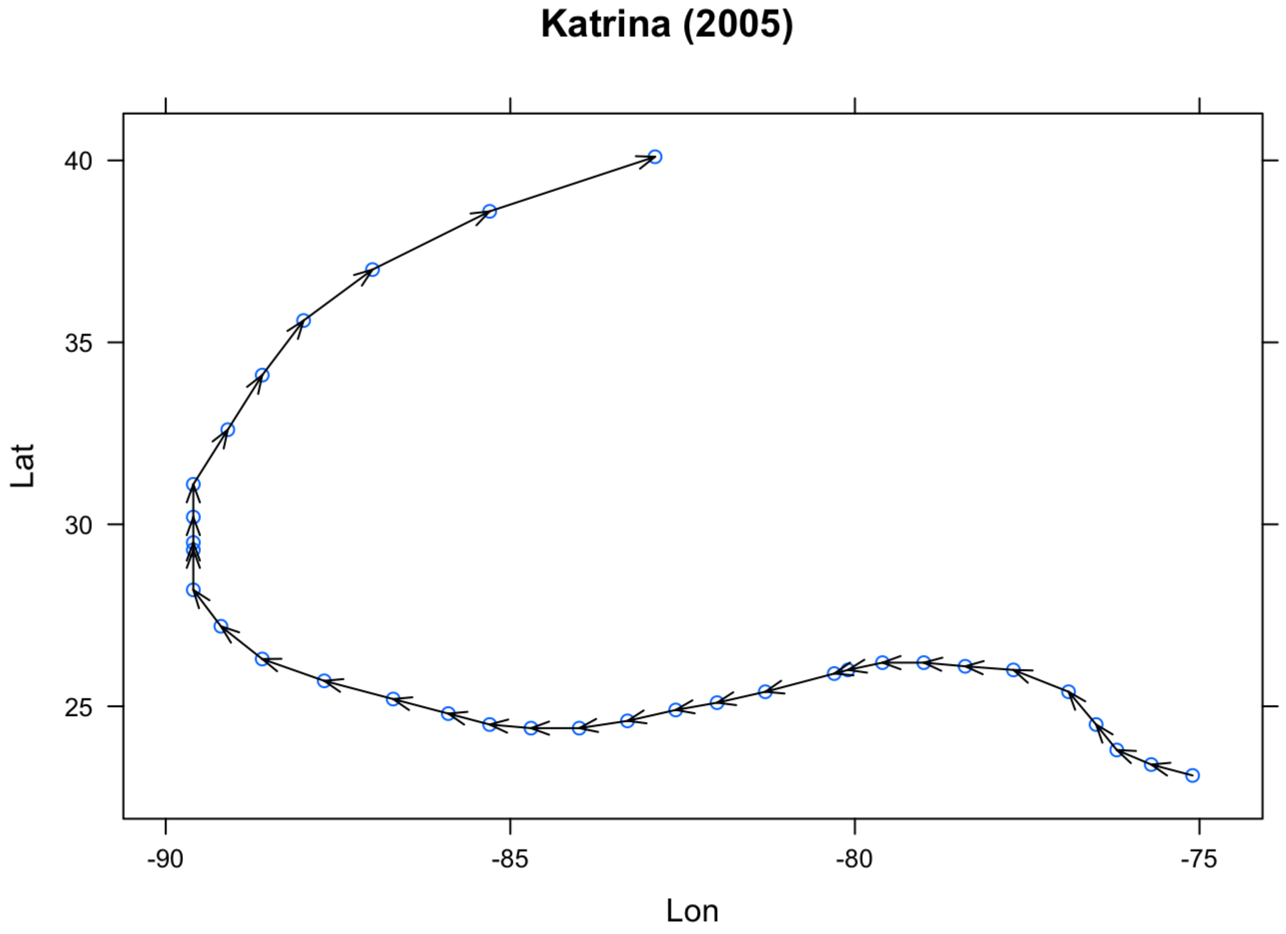
```
histogram( ~Wind | Status,
           data = subset(AL,Year >= 1990),
           nint = 17,
           type = "density",
           endpoints = c(0, 175),
           layout = c(3, 3),
           aspect = 1,
           xlab = "Wind (m/s)",
           panel = function(x,...){
             panel.histogram(x,...)
             panel.mathdensity(dmath = dnorm,
                               col = "red",
                               lwd = 1,
                               args = list(mean=mean(x),
                                           sd=sd(x)), ...)
             }
)
```





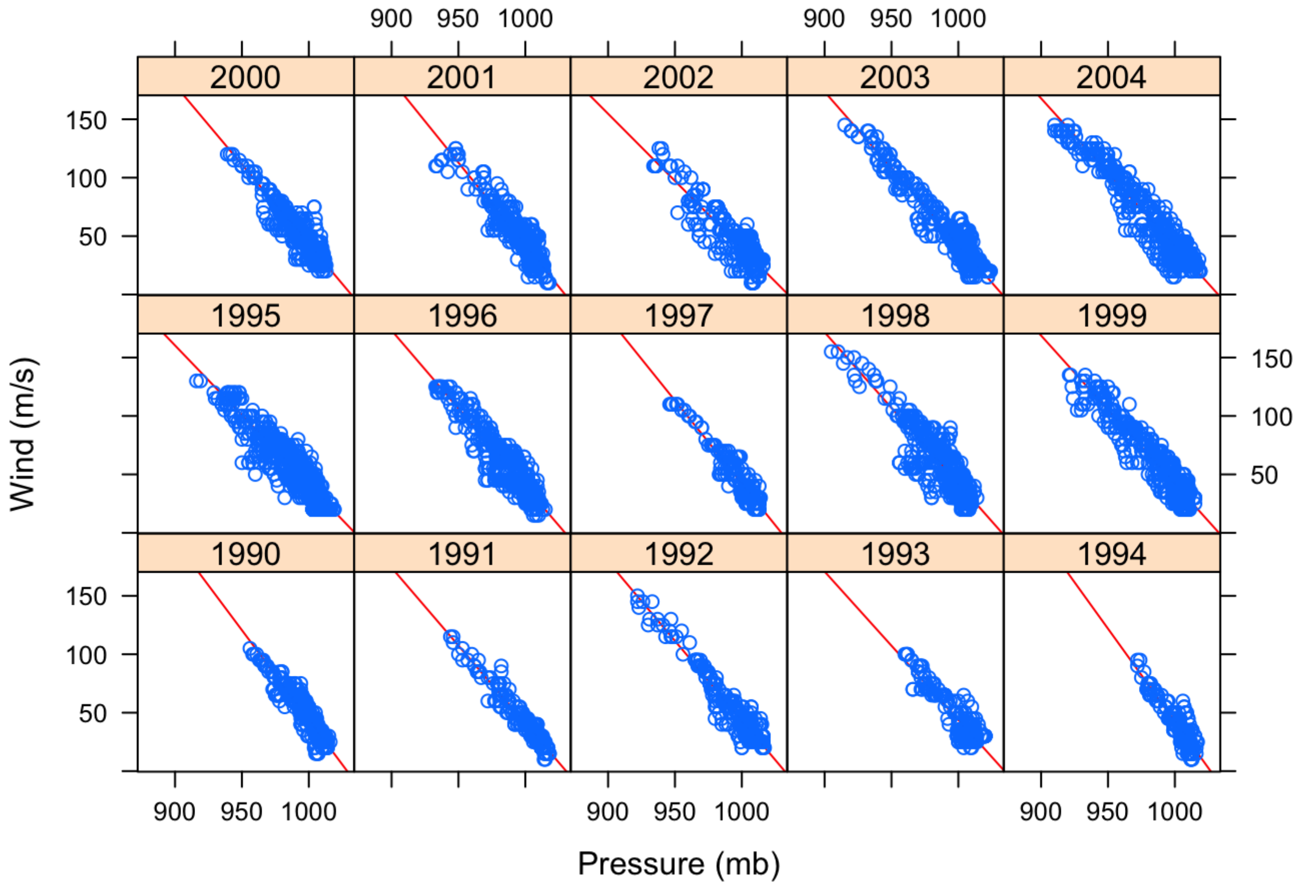
Example 4: Use `panel.arrows` to draw arrows.

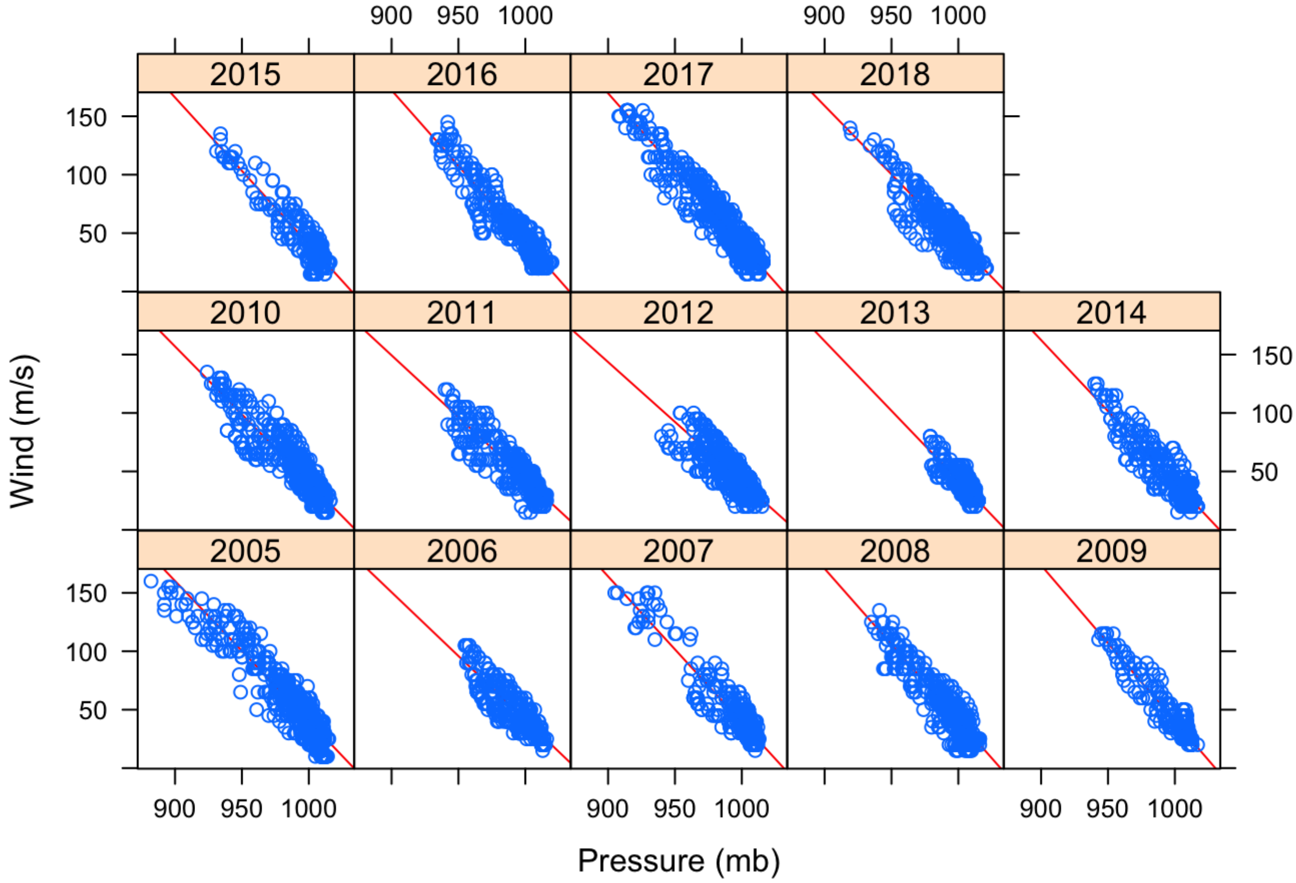
```
xyplot(Lat~Lon,
       data=Kt,
       panel=function(x,y,...){
         panel.xyplot(x,y,...)
         for(i in 1:(length(x)-1)){
           panel.arrows(x[i],y[i],x[i+1],y[i+1],
              angle=20, code=2, length=0.1,...)
         }
       },
       main="Katrina (2005)"
)
```



Example 5: Use `panel.abline` to draw a straight line. Option 1:

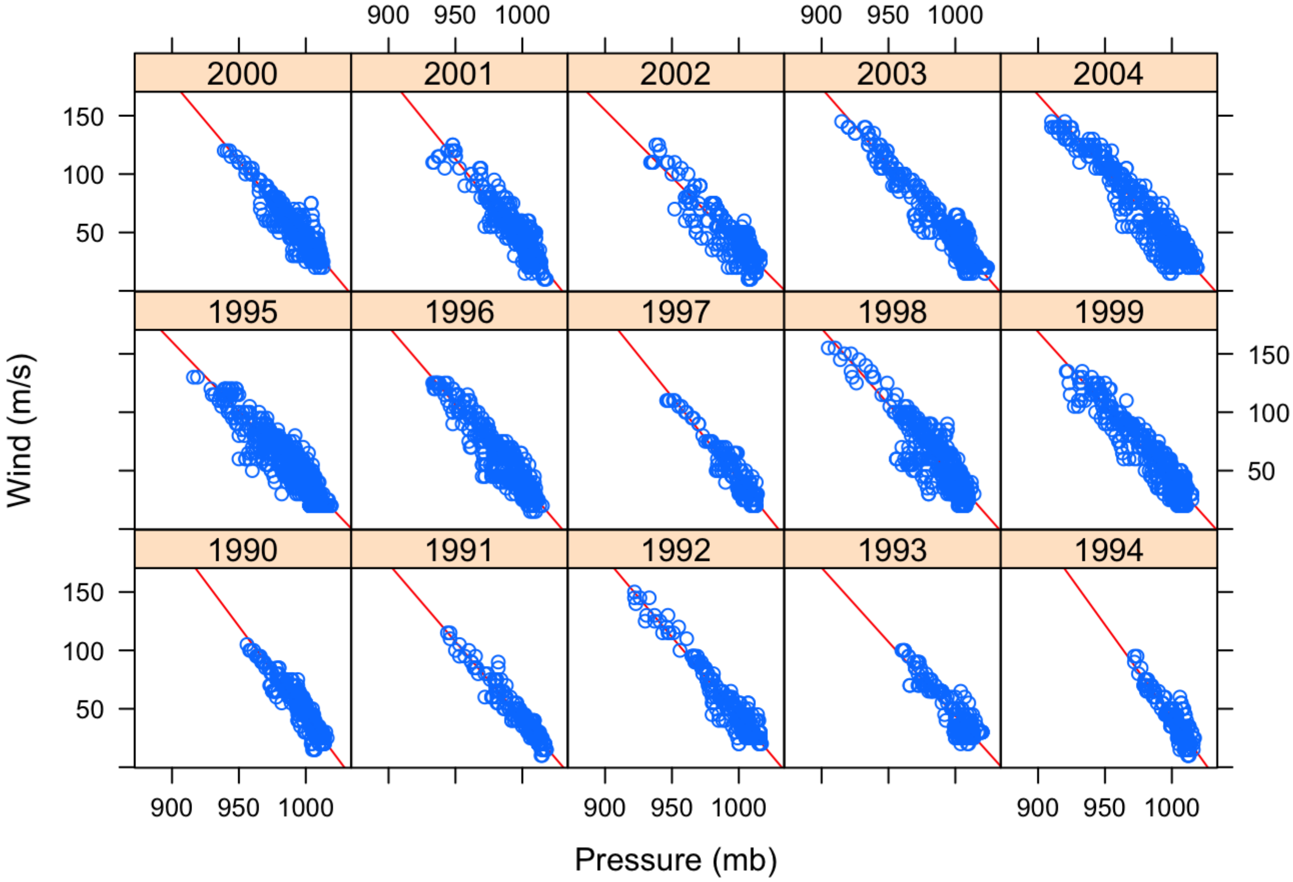
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       panel = function(x,y,...){
         cof <- lm(y~x)$coefficients
         panel.abline(a=cof[1], b=cof[2],
                      col="red", ...)
         panel.xyplot(x,y,...)
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

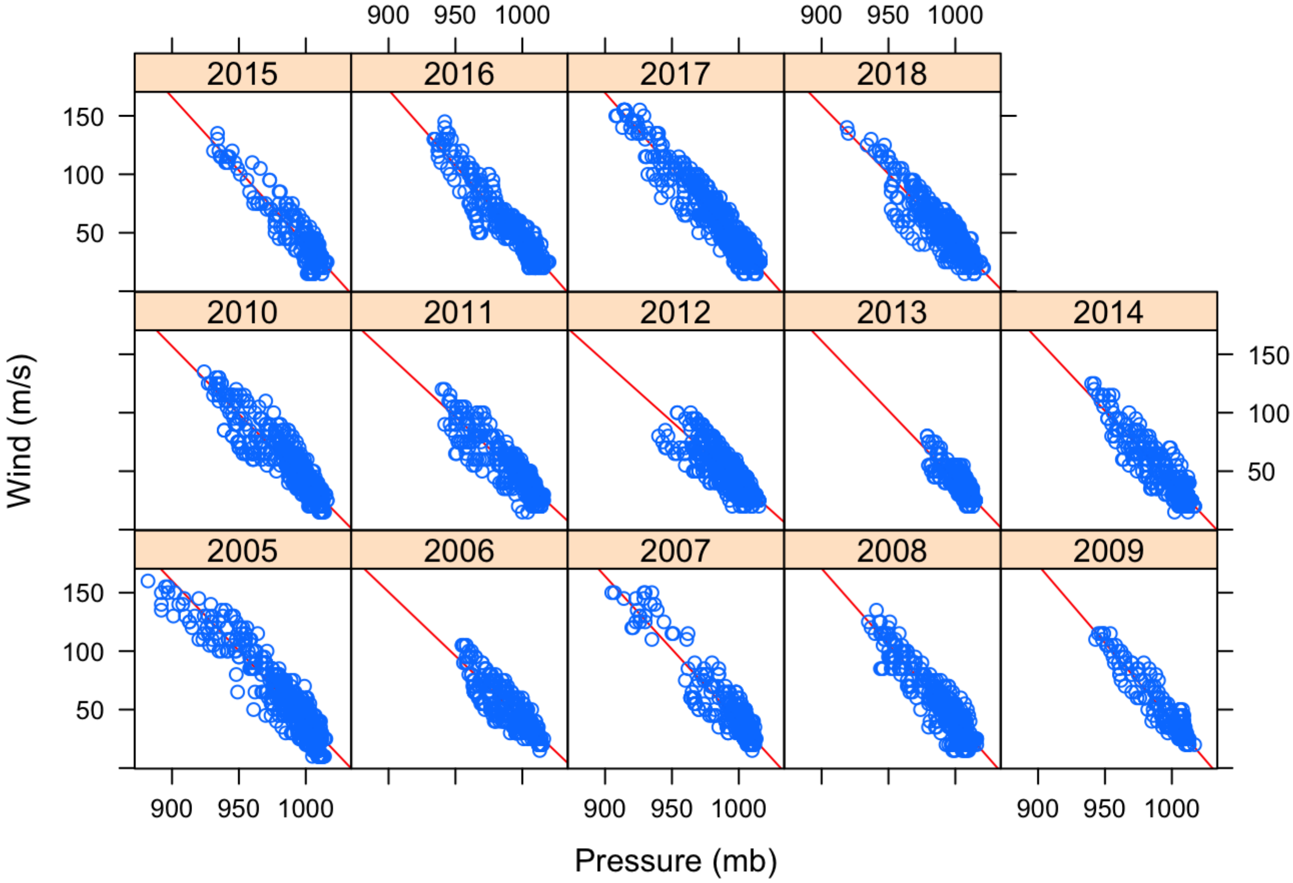

2000
2001
2002
2003
2004
1995
1996
1997
1998
1999
1990
1991
1992
1993
1994
900
950
1000
50
100
150
Wind (m/s)
Pressure (mb)


2015
2016
2017
2018
2010
2011
2012
2013
2014
2005
2006
2007
2008
2009
900
950
1000
50
100
150
Wind (m/s)
Pressure (mb)

Option 2:

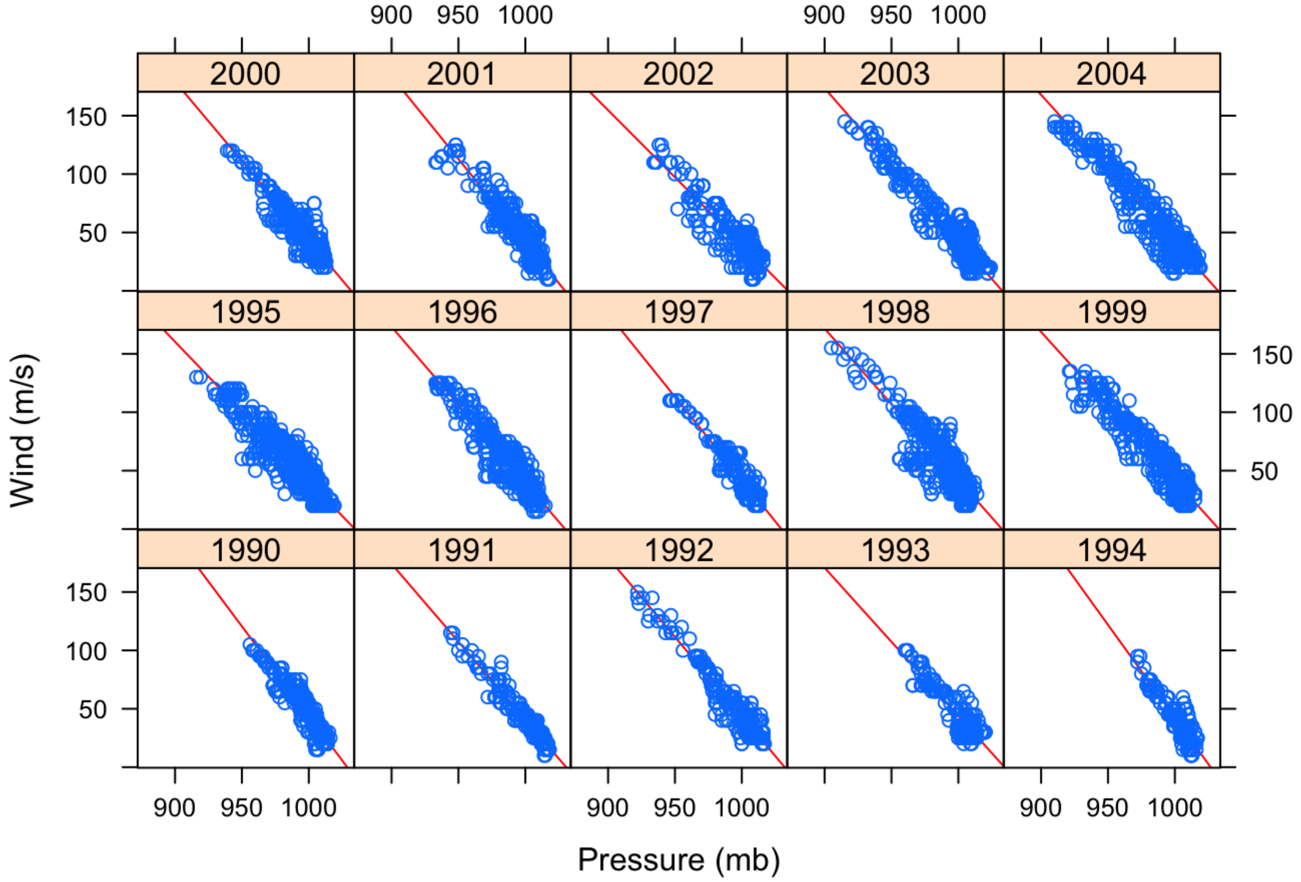
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       panel = function(x,y,...){
         cof <- lm(y~x)
         panel.abline(reg = cof, col="red", ...)
         panel.xyplot(x,y,...)
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

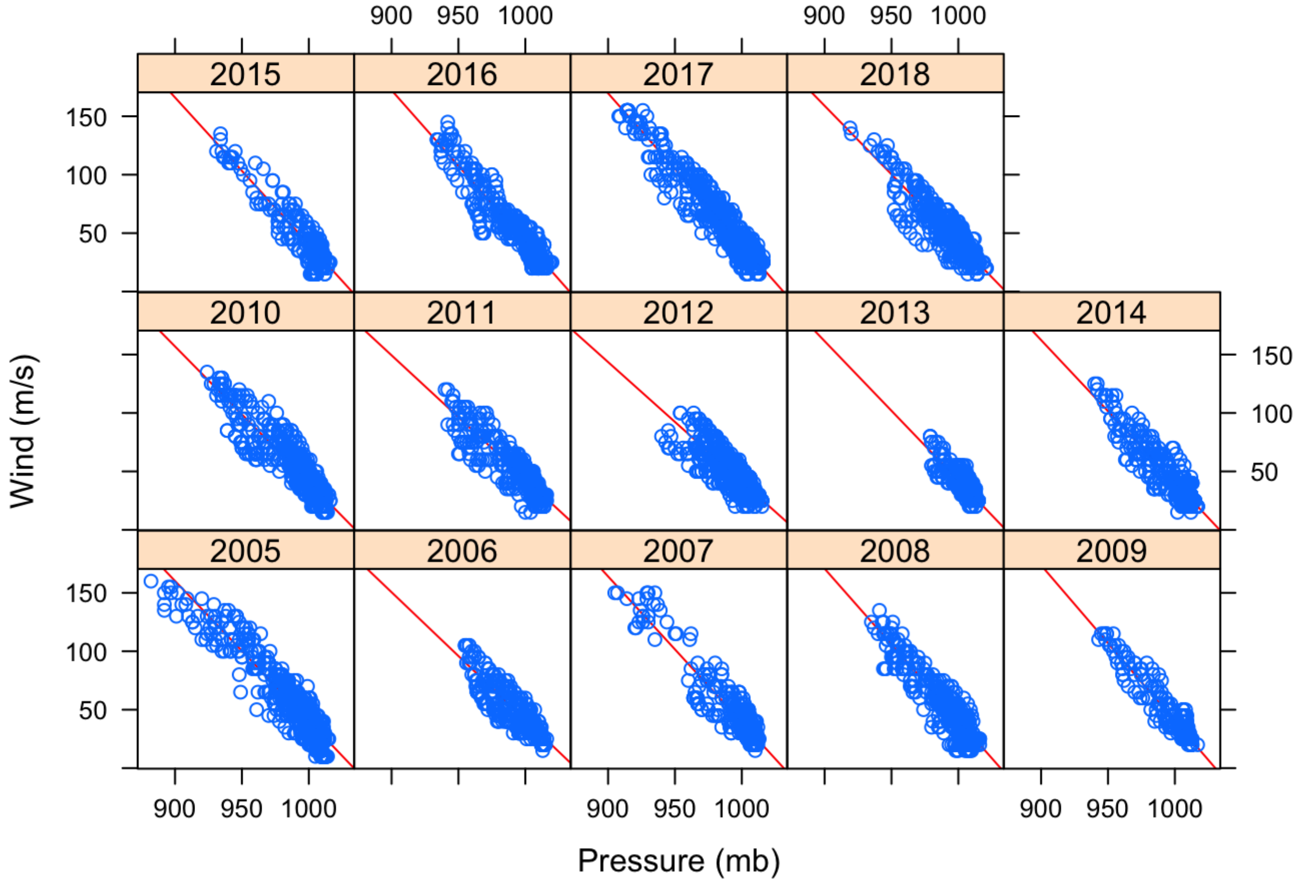

2000
2001
2002
2003
2004
1995
1996
1997
1998
1999
1990
1991
1992
1993
1994
Wind (m/s)
Pressure (mb)
150
100
50
900
950
1000


2015
2016
2017
2018
2010
2011
2012
2013
2014
2005
2006
2007
2008
2009
Wind (m/s)
Pressure (mb)
150
100
50
900
950
1000

Option 3:

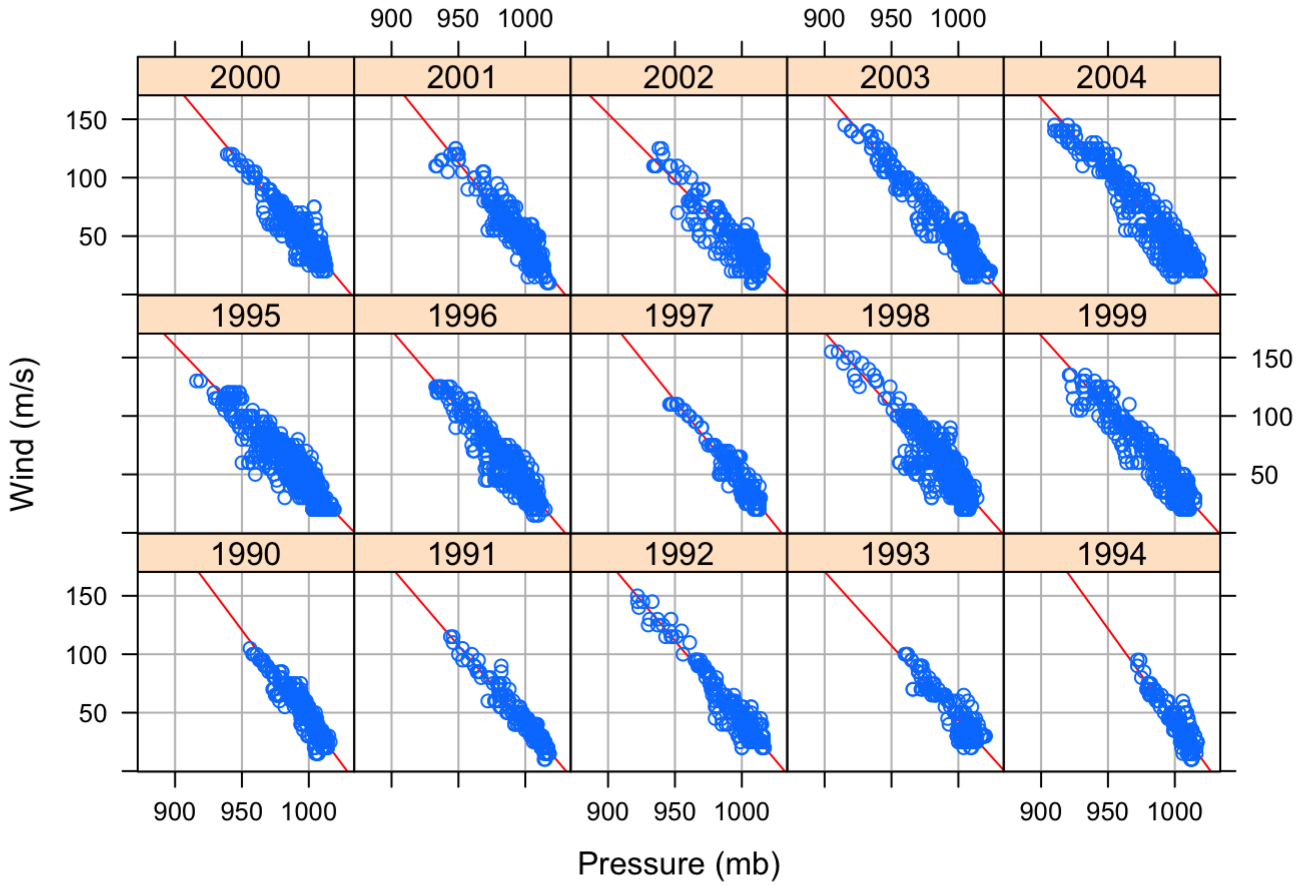
```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       panel = function(x,y,...){
         cof <- lm(y~x)
         panel.lmline(x,y,col="red",...)
         panel.xyplot(x,y,...)
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```

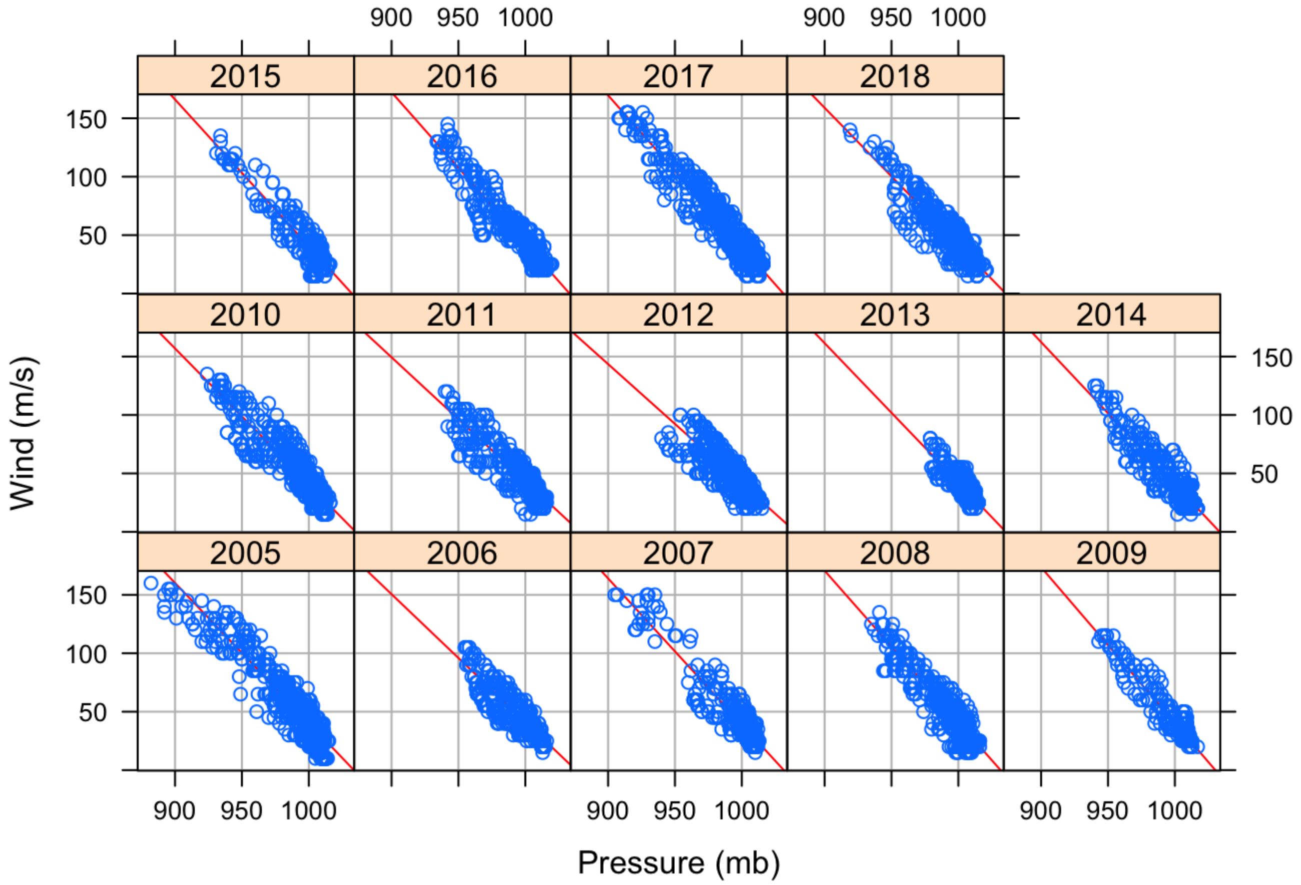




Example 6: Use `panel.grid` to put grid lines over each panel.

```
xyplot(Wind~Pressure | as.factor(Year),
       data=subset(AL,Year >= 1990),
       layout=c(5,3),
       panel = function(x,y,...){
         panel.grid(h=-1, v=-1, col="grey", ...)
         cof <- lm(y~x)
         panel.lmline(x,y,col="red",...)
         panel.xyplot(x,y,...)
         },
       xlab="Pressure (mb)",
       ylab="Wind (m/s)"
       )
```





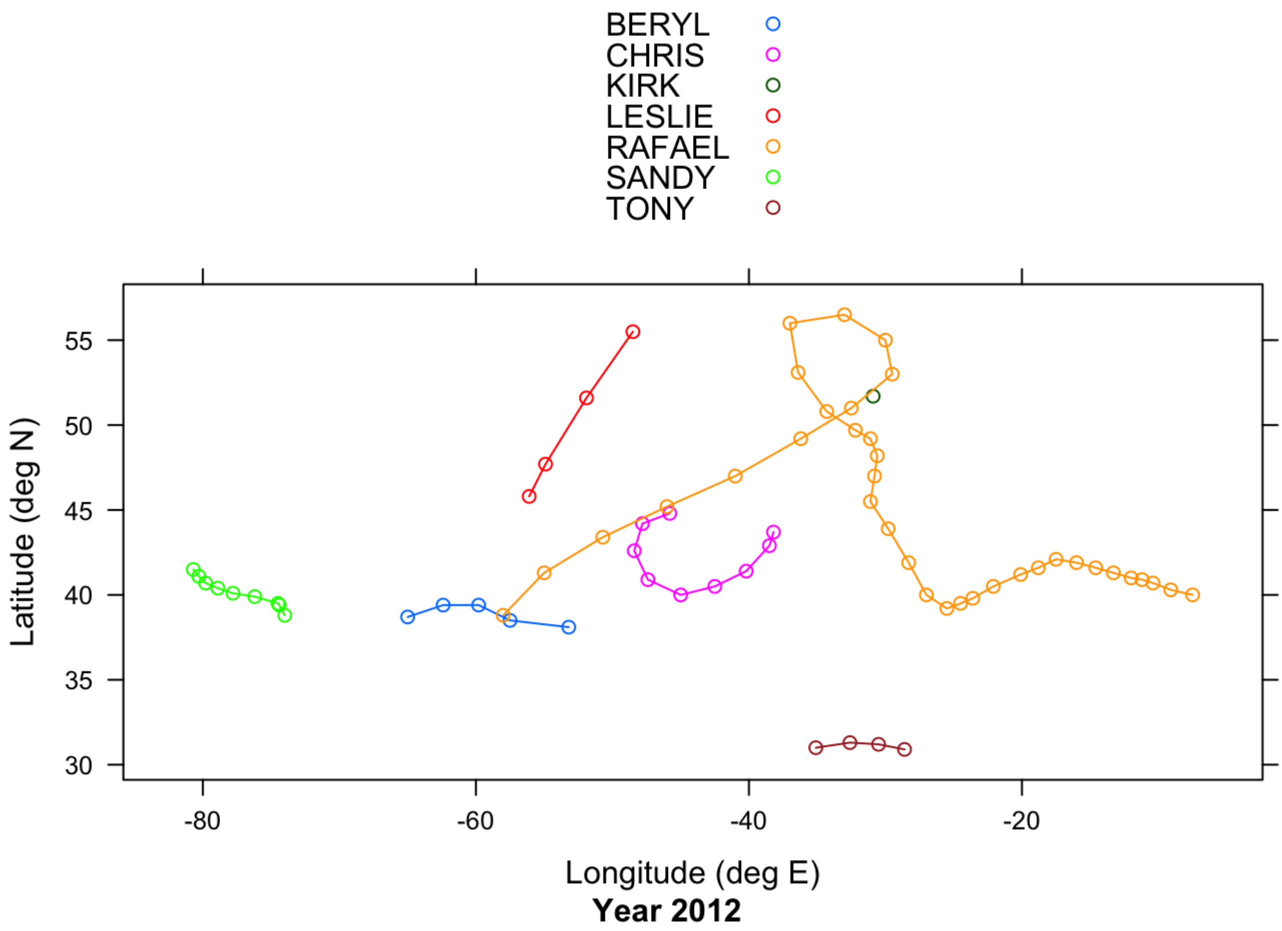
2.3 Use key or legend

2.3.1 Use `auto.key` for fast key generation

It is very common, and desirable, to use a key (legend) when a grouping variable is specified. The `lattice` package uses `groups` to control the grouping and `auto.key` to automatically produce a suitable legend. If `auto.key=TRUE` , a suitable legend will be drawn if a `groups` argument is also provided, and not otherwise.

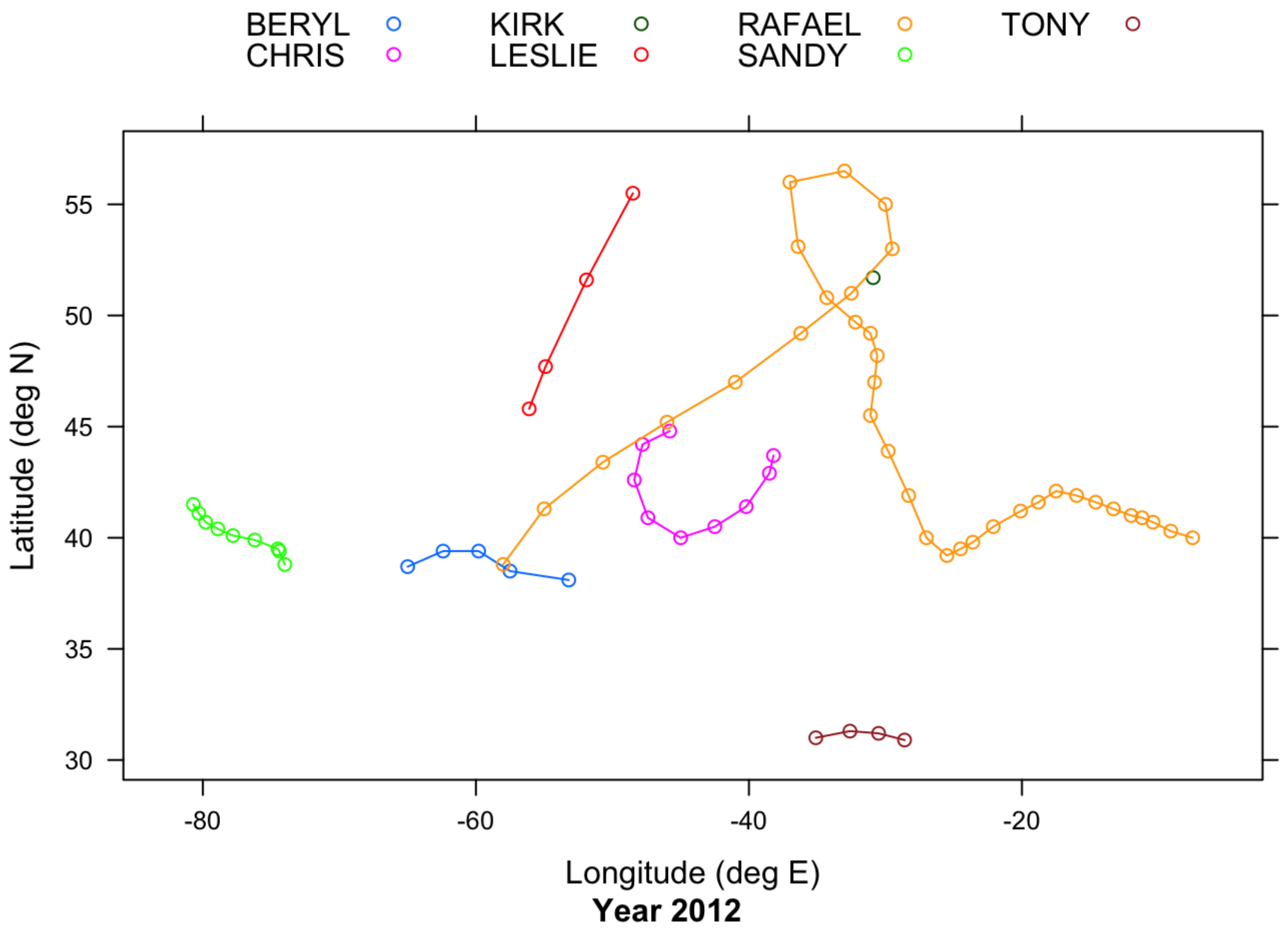
In the following code, the locations of named extra-tropical cyclones in 2012 are grouped by each storm name. A legend is generated above the figure.

```
xyplot(Lat~Lon,
groups = as.factor(Name),
data = subset(AL, Year==2012 & Status=="EX"),
auto.key = TRUE,
type = "b",
scales = list(y=list(tick.number=8)),
xlab="Longitude (deg E)",
ylab="Latitude (deg N)",
sub = "Year 2012"
)
```



We can modify `auto.key` , for example, `auto.key = list(columns=4)` below creates a legend split into four columns.

```
xyplot(Lat~Lon,
groups = as.factor(Name),
data = subset(AL, Year==2012 & Status=="EX"),
auto.key = list(columns=4),
type = "b",
scales = list(y=list(tick.number=8)),
xlab="Longitude (deg E)",
ylab="Latitude (deg N)",
sub = "Year 2012"
)
```



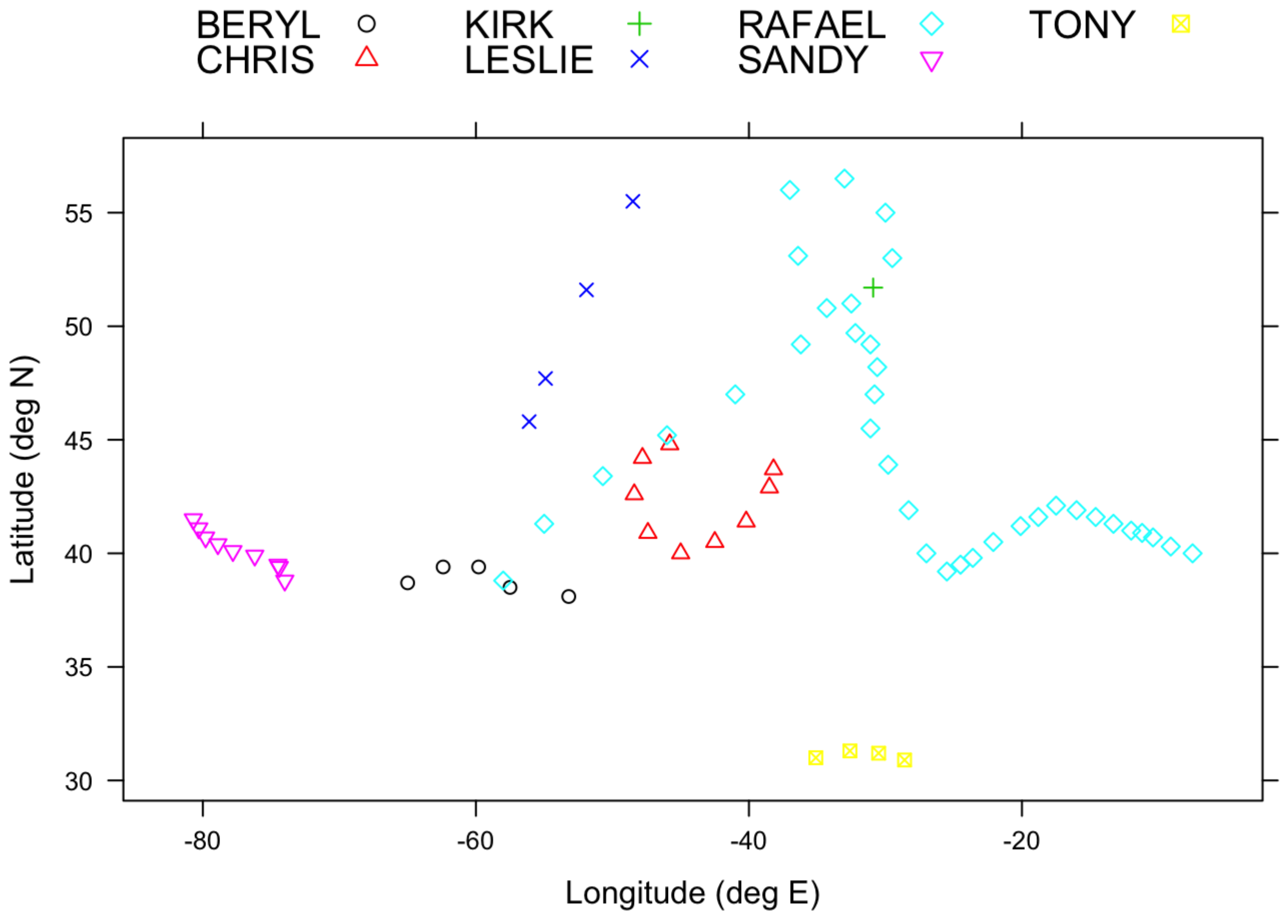
2.3.2 Use `key` for fine-tuned legends

In the following code, the list of arguments for `key` specifies detailed appearance of the legend.

Example 1: The key uses the same group symbols/colors as in the plot.

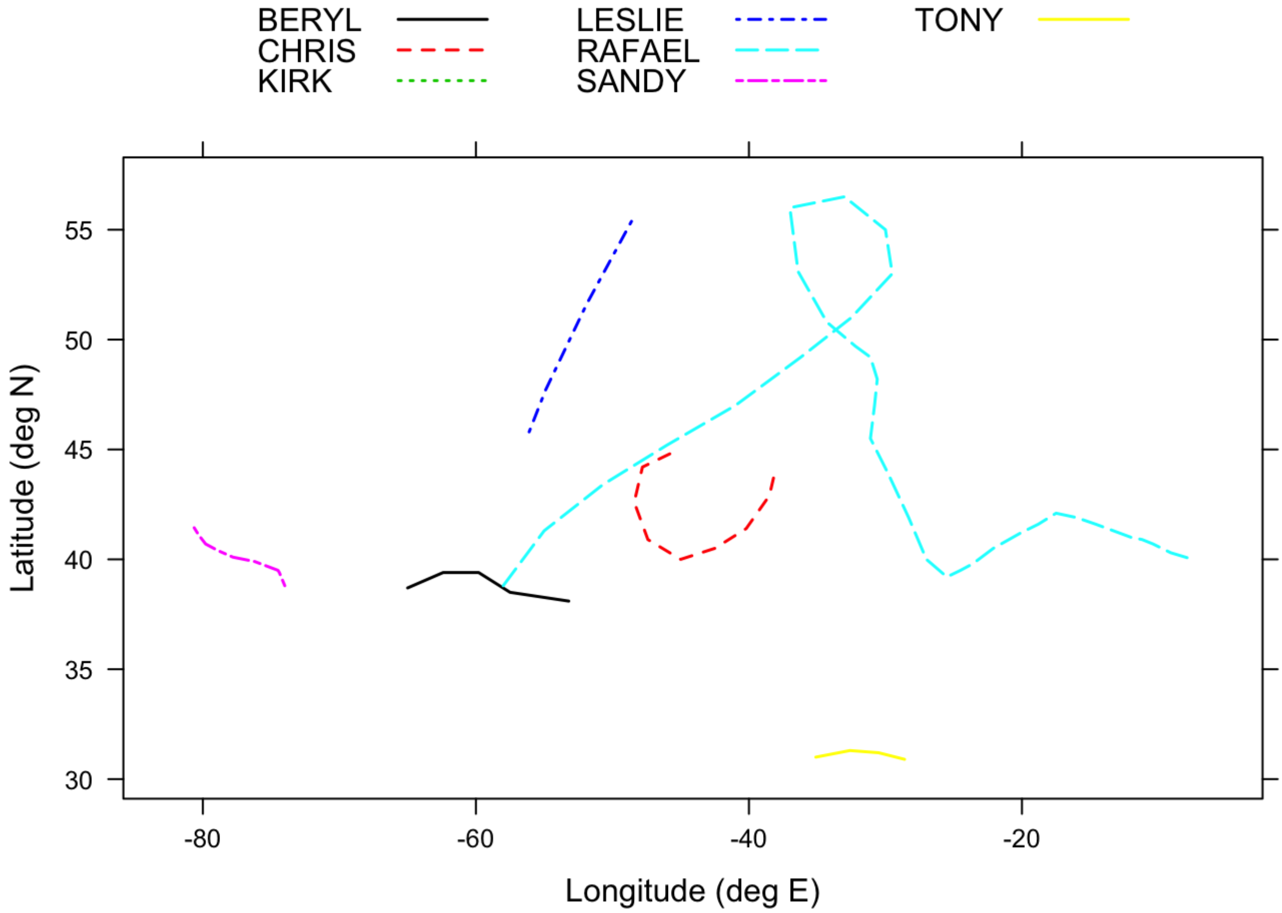
```
namelabel <- levels(as.factor(subset(AL, Year==2012 & Status=="EX")$Name))

xyplot(Lat~Lon,
       groups = as.factor(Name),
       data = subset(AL, Year==2012 & Status=="EX"),
       type = "p",
       pch = 1:7,
       col = 1:7,
       key = list(type = c("p"),
           text = list(label = namelabel, cex = 1.2),
           points = list(col=1:7, pch = 1:7),
           column = 4,
           space = "top"),
       scales = list(y=list(tick.number=8)),
       xlab="Longitude (deg E)",
       ylab="Latitude (deg N)"
)
```



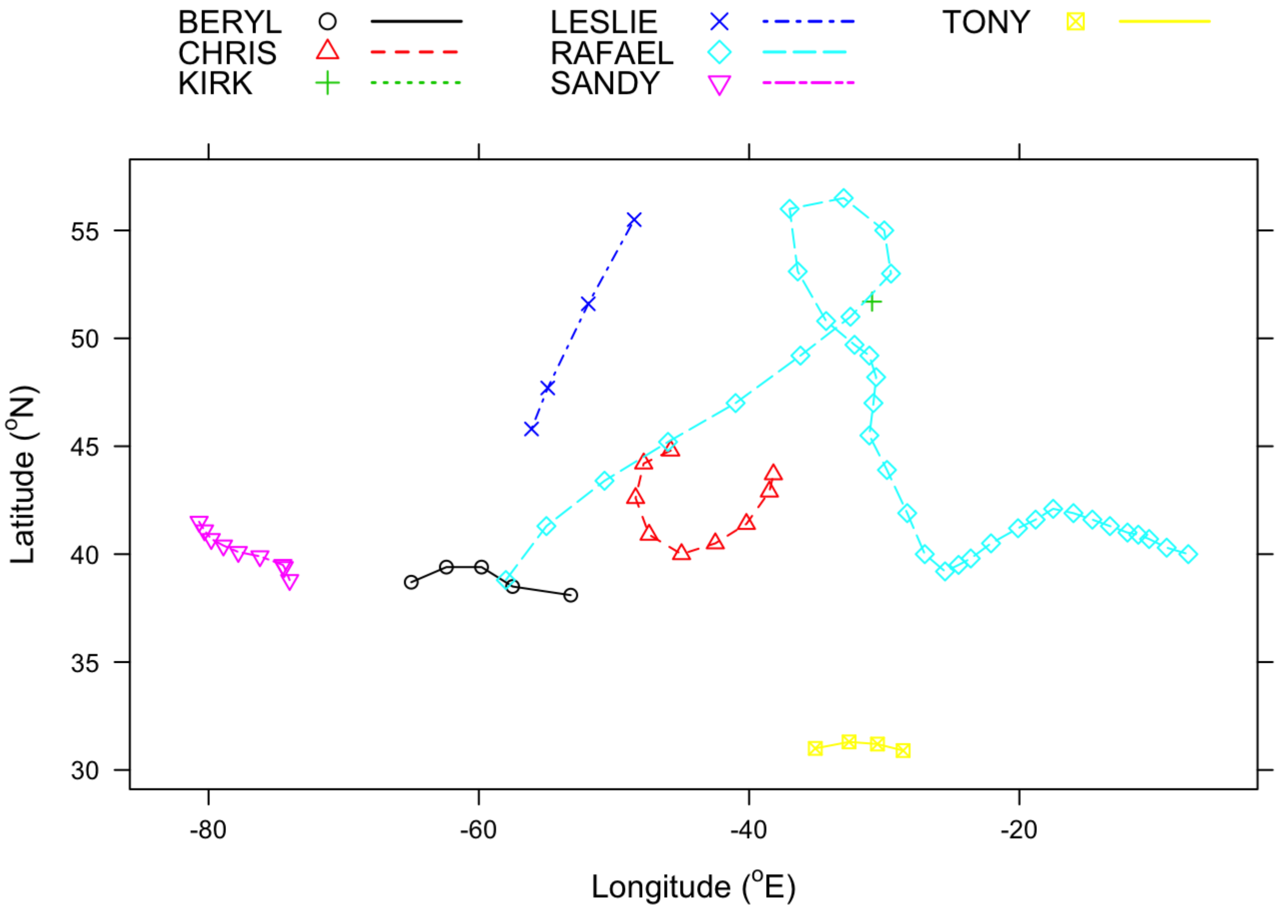
Example 2: The key uses the same group line styles/colors as in the plot.

```
xyplot(Lat~Lon,
       groups = as.factor(Name),
       data = subset(AL, Year==2012 & Status=="EX"),
       type = "l",
       lty = 1:7,
       lwd = 1.5,
       col = 1:7,
       key = list(type = "l",
           text = list(label = namelabel),
           lines = list(col=1:7, lty= 1:7, lwd=1.5),
           column = 3,
           space = "top"),
       scales = list(y=list(tick.number=8)),
       xlab="Longitude (deg E)",
       ylab="Latitude (deg N)"
)
```



Example 3: The key uses the same group symbols as well as line styles/colors as in the plot.

```
xyplot(Lat~Lon,
       groups = as.factor(Name),
       data = subset(AL, Year==2012 & Status=="EX"),
       type = "b",
       lty = 1:7,
       pch = 1:7,
       col = 1:7,
       key = list(type = c("l"),
           text = list(label = namelabel),
           points = list(col=1:7, pch = 1:7),
           lines = list(col=1:7, lty= 1:7, lwd=1.5),
           column = 3,
           space = "top"),
       scales = list(y=list(tick.number=8)),
       xlab=expression(paste("Longitude (", {}^o,"E)")),
       ylab=expression(paste("Latitude (", {}^o,"N)"))
)
```



2.4 Overlay continental outlines

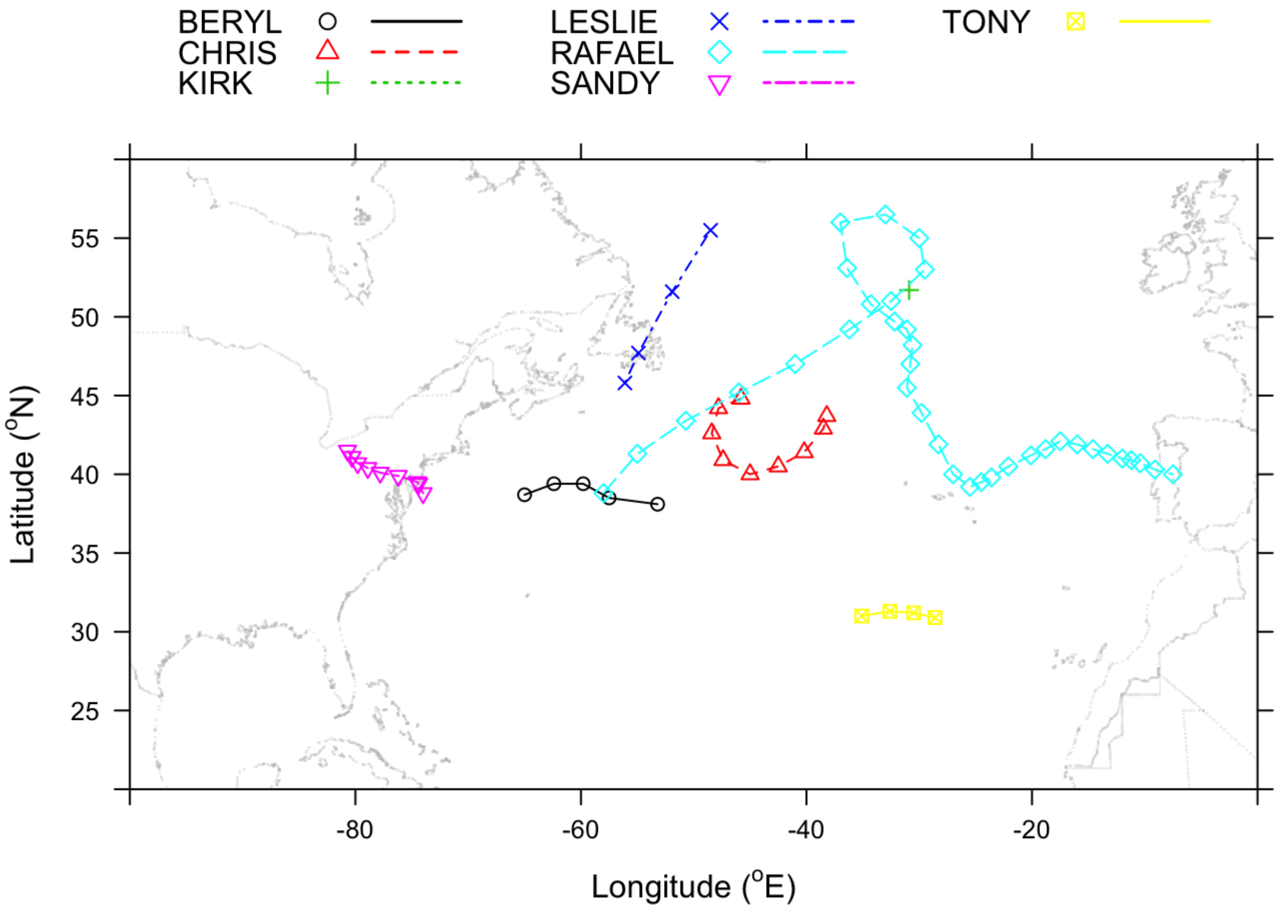
It is often desirable to overlay our data visualization with a map or continental outlines for geographical information. The following code uses the `maps` library, although it is not the only method. Note how the `ylim` and `xlim` are consistently assigned for `mp` in `panal.map` and in the `xyplot` .

```
library(maps)
```

```
## Warning: package 'maps' was built under R version 3.4.4
```

```
mp <- map("world", ylim=c(20,60), xlim=c(-100, 0),plot = FALSE, fill=FALSE)
panel.map <- function(x,y,...)
{panel.xyplot(mp$x, mp$y,pch=".",col="gray",alpha=0.25)
 panel.xyplot(x,y,...)
}

xplot(Lat~Lon,
       groups = as.factor(Name),
       data = subset(AL, Year==2012 & Status=="EX"),
       type = "b",
       lty = 1:7,
       pch = 1:7,
       col = 1:7,
       key = list(type = c("l"),
           text = list(label = namelabel),
           points = list(col=1:7, pch = 1:7),
           lines = list(col=1:7, lty= 1:7, lwd=1.5),
           column = 3,
           space = "top"),
       scales = list(y=list(tick.number=8)),
       xlab=expression(paste("Longitude (", {}^o,"E)")),
       ylab=expression(paste("Latitude (", {}^o,"N)")),
       ylim=c(20,60), xlim=c(-100, 0),
       panel=panel.map
)
```



Question 13: Create multipanel plots of the tracks (`Lat` vs. `Lon`) by storm name `Name` in `Year` 2005 using `xyplot` . Overlay each panel with a map. Note that the command pair `trellis.device` and `dev.off()` will output this figure into a pdf file in your working directory instead of showing it on your screen.

```
#### Answers
trellis.device(pdf,file="Question12.pdf")

#### Write your code for visiaulzation between these two lines.

dev.off()
```