CWRU DSCI351-351m-451: Exploratory Multi-variate Pair-wise Correlation Plots

Profs: R. H. French, L. S. Bruckman, P. Leu, K. Davis, S. Cirlos

TAs: W. Oltjen, K. Hernandez, M. Li, M. Li, D. Colvin

11 October, 2022

Contents

5.2.3 Pair Coding

- Reading posted in the Class Repo
- What is Code Review
- 11 Best Practices for Peer Code Review

5.2.4 Everything is a variable

And in EDA

- Finding relationships among variables
  - Starting with scatterplots
- And continuing with linear correlations
  - Is a good way to go

Pairs plots are a fast way to EDA for relationships

- These may be expected, or unexpected
- They don't necessarily mean causality

5.2.5 Scatter Plot Matrices, Pair-wise Correlation "Pairs" plots

5.2.5.1 Using the iris dataset in R as an example

Lets load the iris dataset, check out its background

And then look at correlation coefficients among variables: numerically

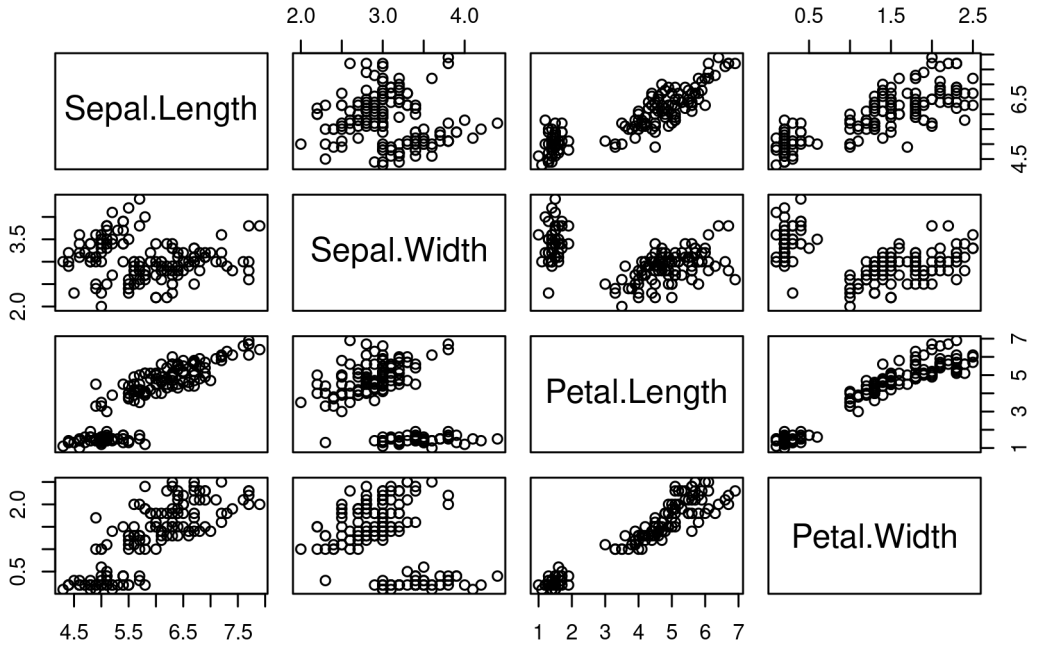
```
data(iris)
?iris
cor(iris[,1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
## Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

Tabular data doesn't communicate to us very well

```
pairs(iris[,1:4])
```

5.2.5.1.1 so lets use a pairwise linear correlation plot



This is a nice example of un-biased analytics

- We can visually see if relationships are present
- but not necessarily what their origin or nature is

The upper right and lower left quadrants are identical

- the diagonal is the variable names
- I find it best to read the lower left quadrant

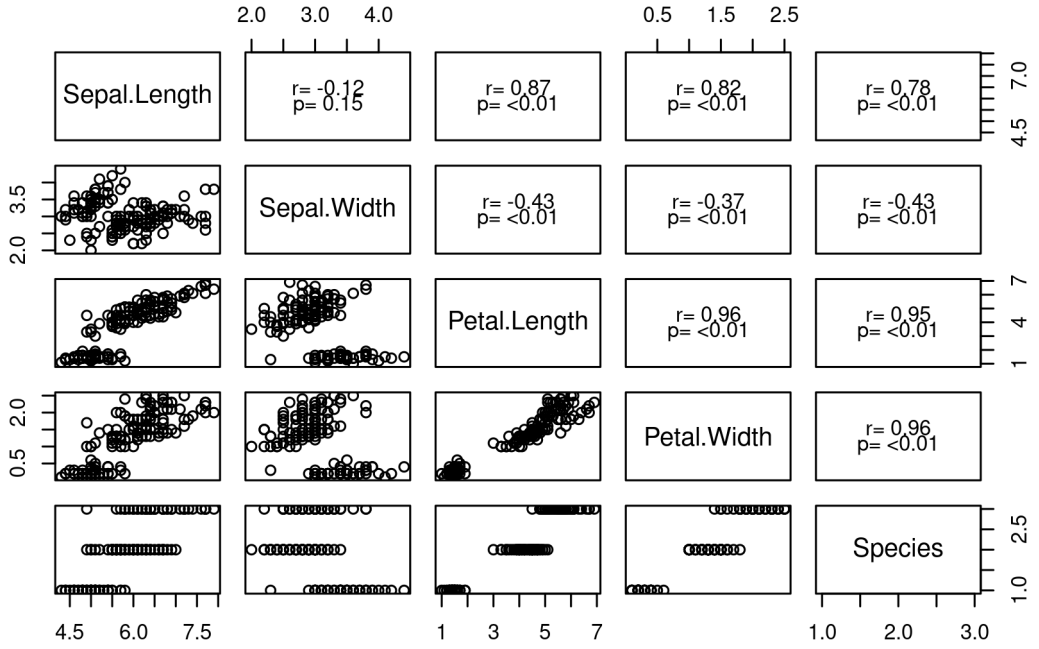
5.2.5.1.2 Lets make a better pairs plot

With the corrrelation coefficients and p values

- make r = correlation coefficients
- make p = p values for the correlation test
- and lets make this into a function we can use later also.

```
panel.cor <- function(x, y, digits = 2, cex.cor, ...)
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  # correlation coefficient
  r <- cor(x, y)
  txt <- format(c(r, 0.123456789), digits = digits)[1]
  txt <- paste("r= ", txt, sep = "")
  text(0.5, 0.6, txt)

  # p-value calculation
  p <- cor.test(x, y)$p.value
  txt2 <- format(c(p, 0.123456789), digits = digits)[1]
  txt2 <- paste("p= ", txt2, sep = "")
  if (p < 0.01) txt2 <- paste("p= ", "<0.01", sep = "")
  text(0.5, 0.4, txt2)
}

pairs(iris, upper.panel = panel.cor)
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```



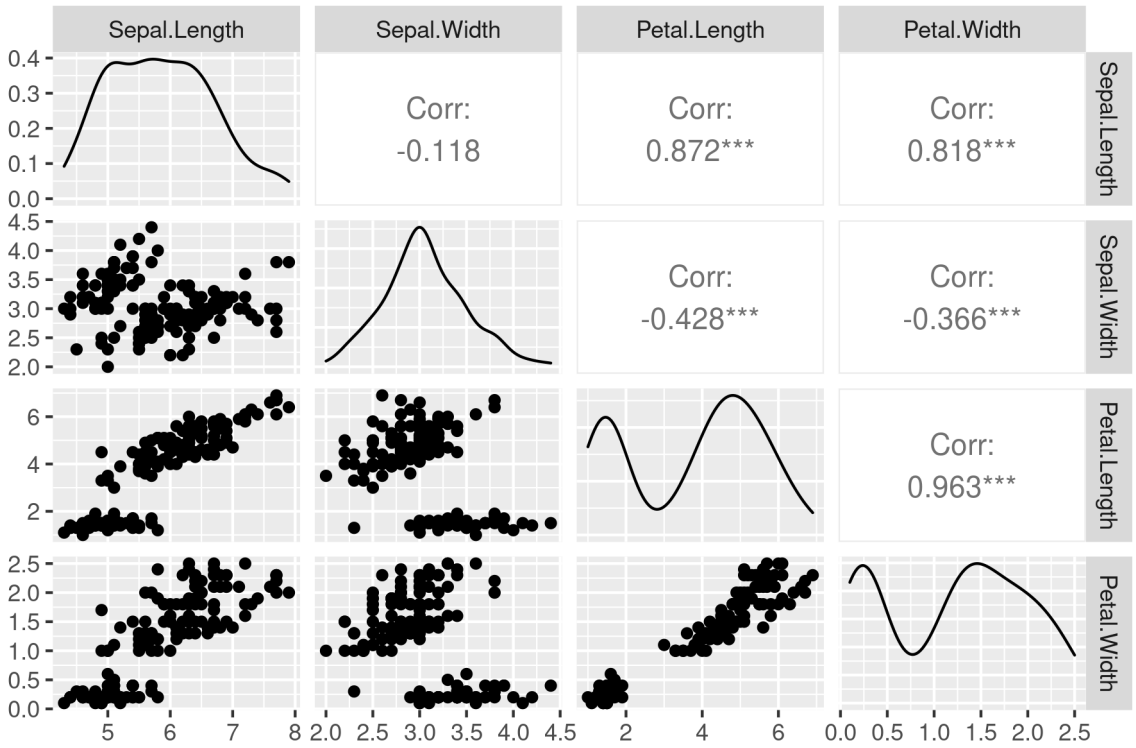

```
library(GGally)
```

5.2.5.1.3 even easier to do with ggplot and GGally

```
## Loading required package: ggplot2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(iris[,1:4])
```



Some tuning of the x-axis labels required!

5.2.5.2 Lets take another run at scatterplots

Lets use the mtcars dataset in R

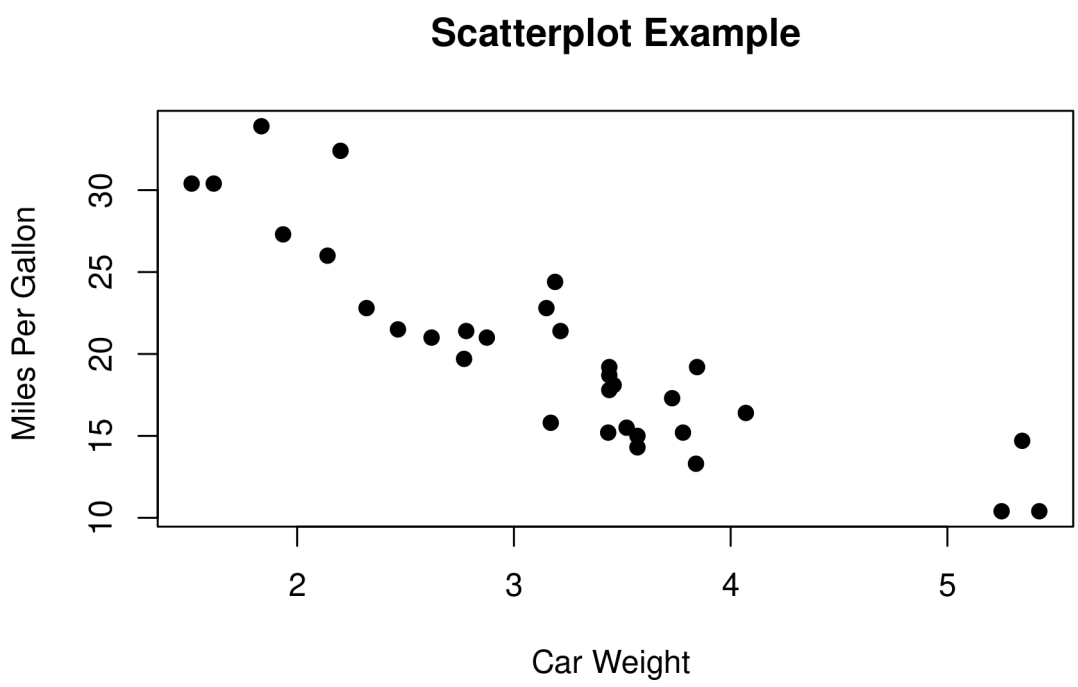
- Motor Trend Car Road Tests for 32, 1973-4 models

```
attach(mtcars)
```

5.2.5.2.1 Simple Scatterplots

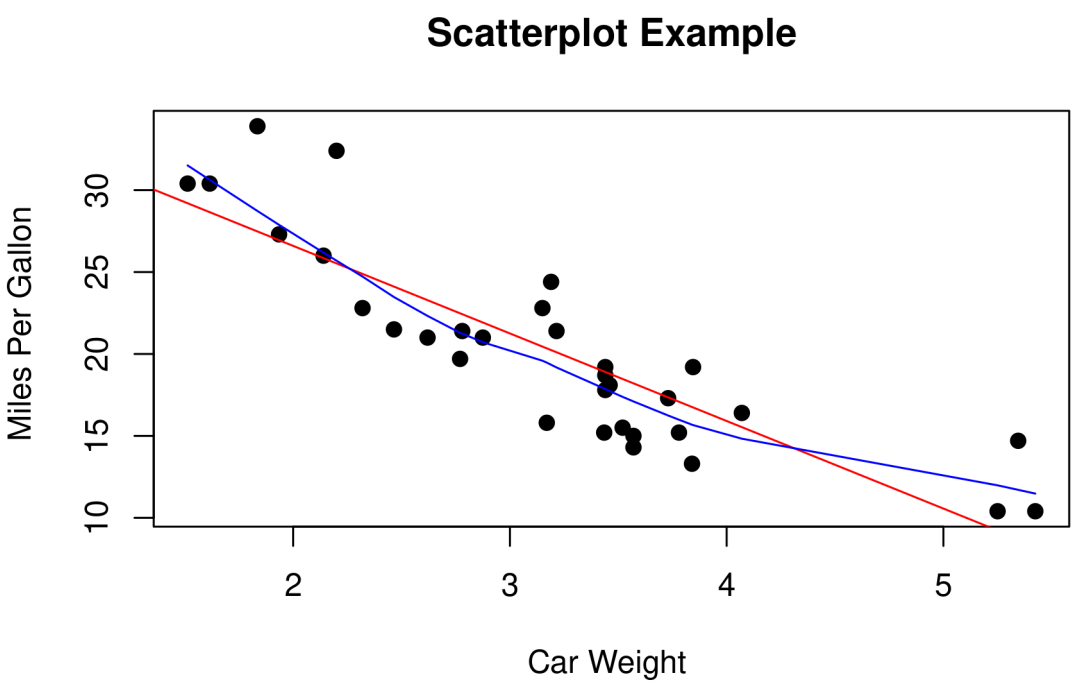
```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
?mtcars
plot(wt, mpg, main = "Scatterplot Example",
     xlab = "Car Weight ", ylab = "Miles Per Gallon ", pch = 19)
```



Add fit lines

```
# Add fit lines
plot(wt, mpg, main = "Scatterplot Example",
    xlab = "Car Weight ", ylab = "Miles Per Gallon ", pch = 19)
abline(lm(mpg~wt), col = "red") # regression line (y~x)
lines(lowess(wt,mpg), col = "blue") # lowess line (x,y)
```



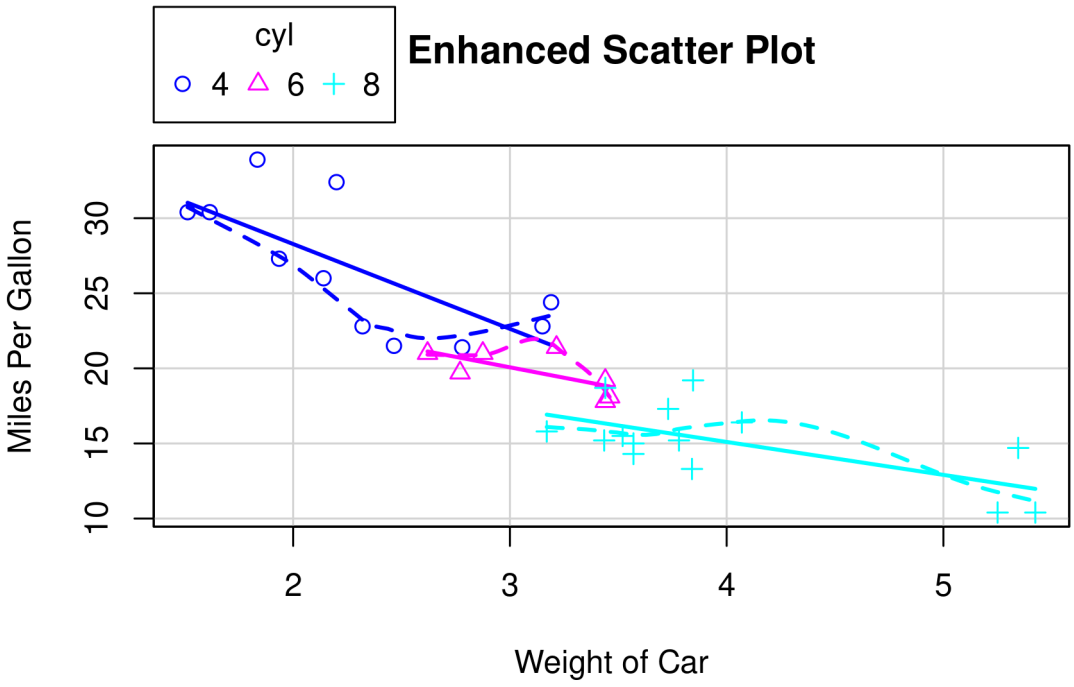
Try scatterplot function in the car package

```
# Enhanced Scatterplot of MPG vs. Weight
# by Number of Car Cylinders
library(car)
```

```
## Loading required package: carData
```

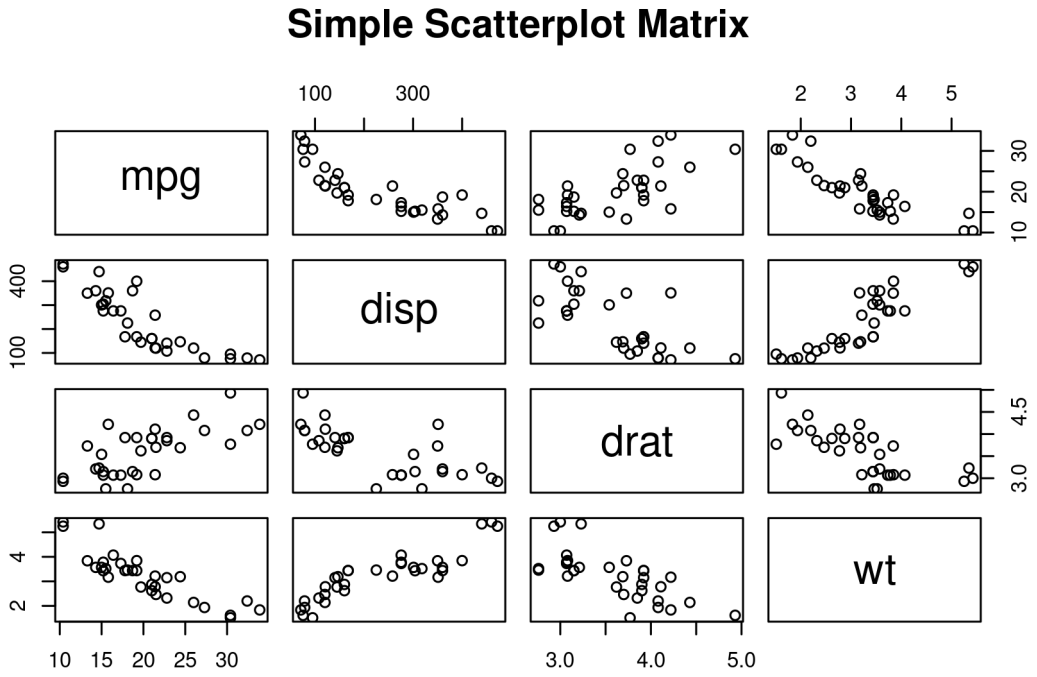
```
??car
scatterplot(mpg ~ wt | cyl, data = mtcars, xlab = "Weight of Car",
            ylab = "Miles Per Gallon", main = "Enhanced Scatter Plot",
            legend = row.names(mtcars))
```

```
## Warning in applyDefaults(legend, defaults = list(), type = "legend"): unnamed
## legend arguments, will be ignored
```




```
# Basic Scatterplot Matrix
pairs(~mpg+disp+drat+wt,data = mtcars,
      main = "Simple Scatterplot Matrix")
```

5.2.5.3 Now onto pairs plots, i.e. scatterplot matrices



and using the car package

```
# Scatterplot Matrices from the car Package
library(car)
attach(mtcars)
```

```
## The following objects are masked from mtcars (pos = 5):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

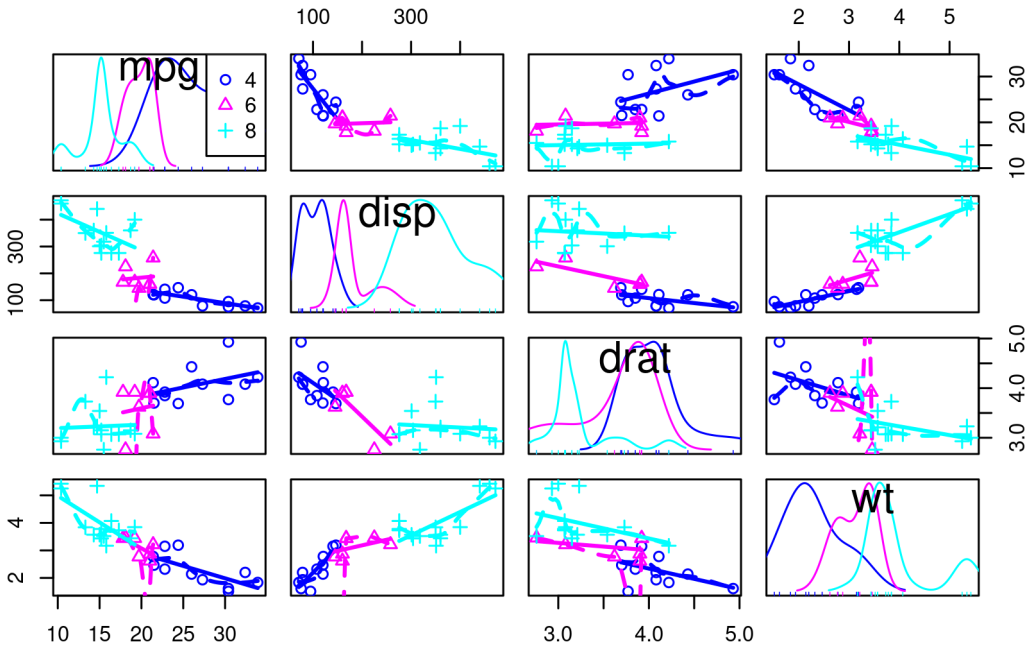
```
scatterplotMatrix(~ mpg + disp + drat + wt | cyl, data = mtcars,
                  legend = "Three Cylinder Options")
```

```
## Warning in applyDefaults(legend, defaults = list(coords = NULL, pt.cex = cex, :
## unnamed legend arguments, will be ignored
```

```
## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth
```

```
## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth
```

```
## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit smooth
```



and the gclus package

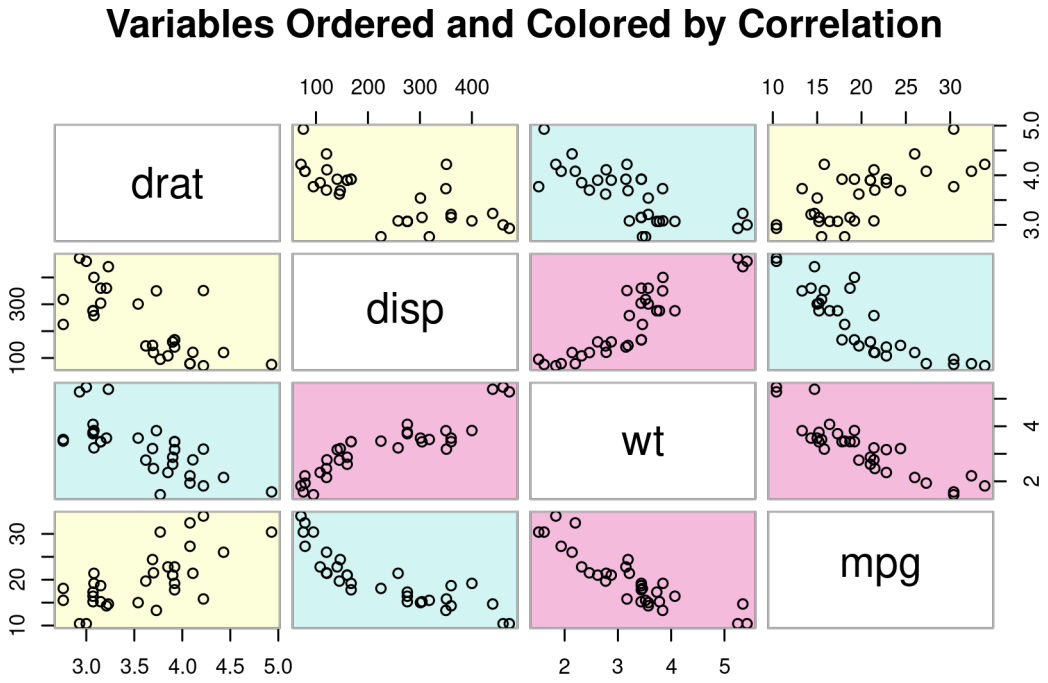
The gclus package gives pairs plots

- colored by magnitude of the correlation coefficient
- very useful "signatures"

```
# Scatterplot Matrices from the gclus Package
library(gclus)
```

```
## Loading required package: cluster
```

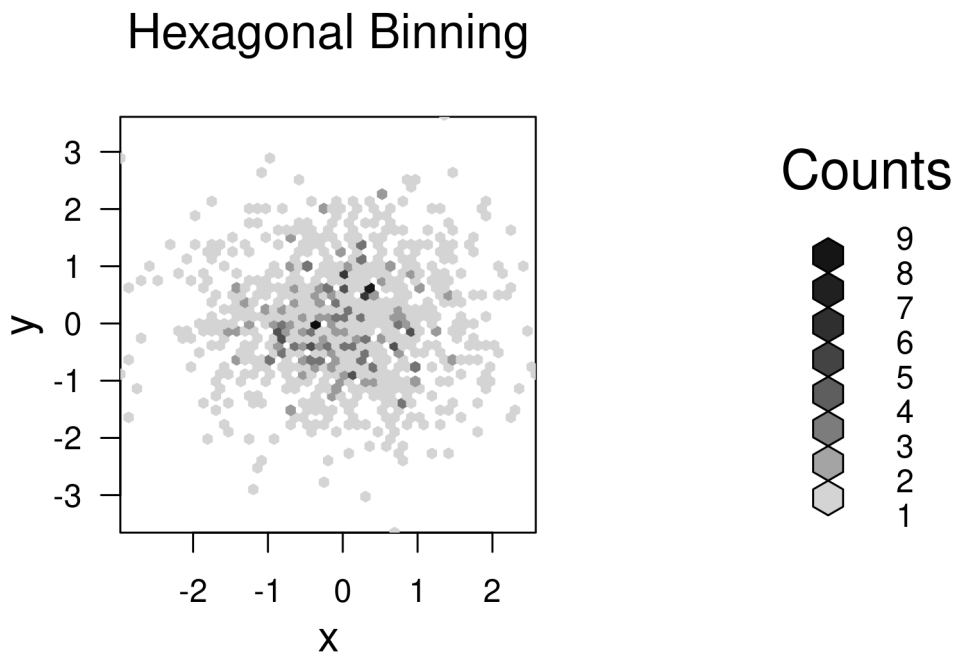
```
??gclus
dta <- mtcars[c(1,3,5,6)] # get data
dta.r <- abs(cor(dta)) # get correlations
dta.col <- dmat.color(dta.r) # get colors
# reorder variables so those with highest correlation
# are closest to the diagonal
dta.o <- order.single(dta.r)
cpairs(dta, dta.o, panel.colors = dta.col, gap = 0.5,
       main = "Variables Ordered and Colored by Correlation" )
```



5.2.5.4 High Density scatterplots with Binning

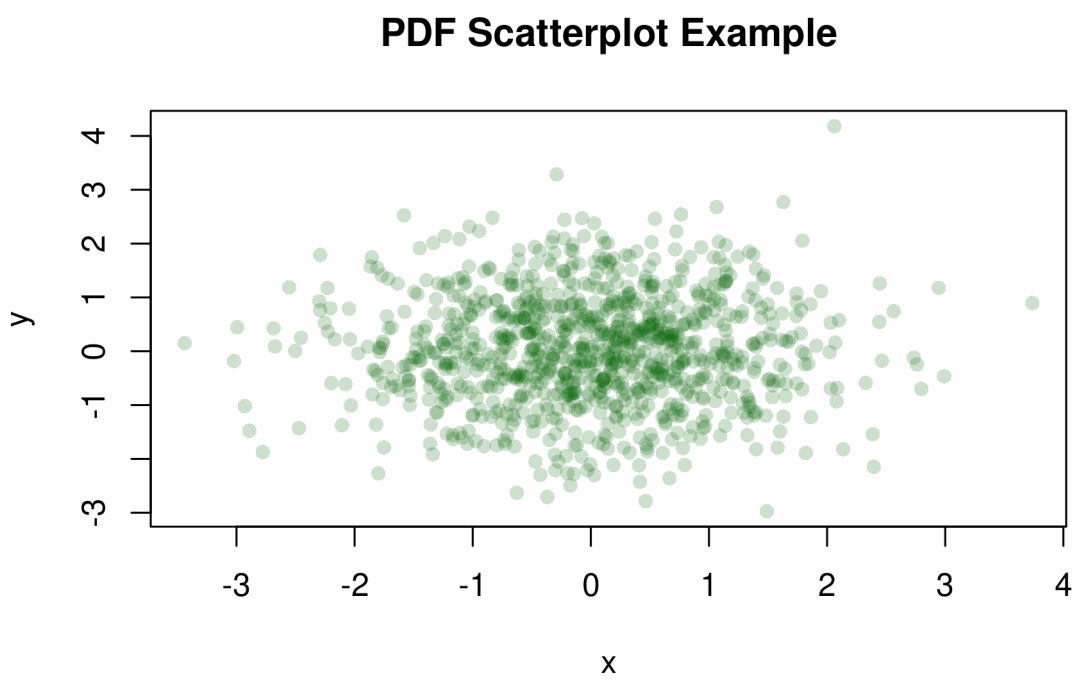
```
# High Density Scatterplot with Binning
library(hexbin)
x <- rnorm(1000)
y <- rnorm(1000)
bin <- hexbin(x, y, xbins = 50)
plot(bin, main = "Hexagonal Binning")
```

5.2.5.4.1 using hexbin package




```
# High Density Scatterplot with Color Transparency
x <- rnorm(1000)
y <- rnorm(1000)
plot(x,y, main = "PDF Scatterplot Example",
     col = rgb(0,100,0,50,maxColorValue = 255), pch = 16)
```

5.2.5.4.2 with sunflowerpot if the points overlap

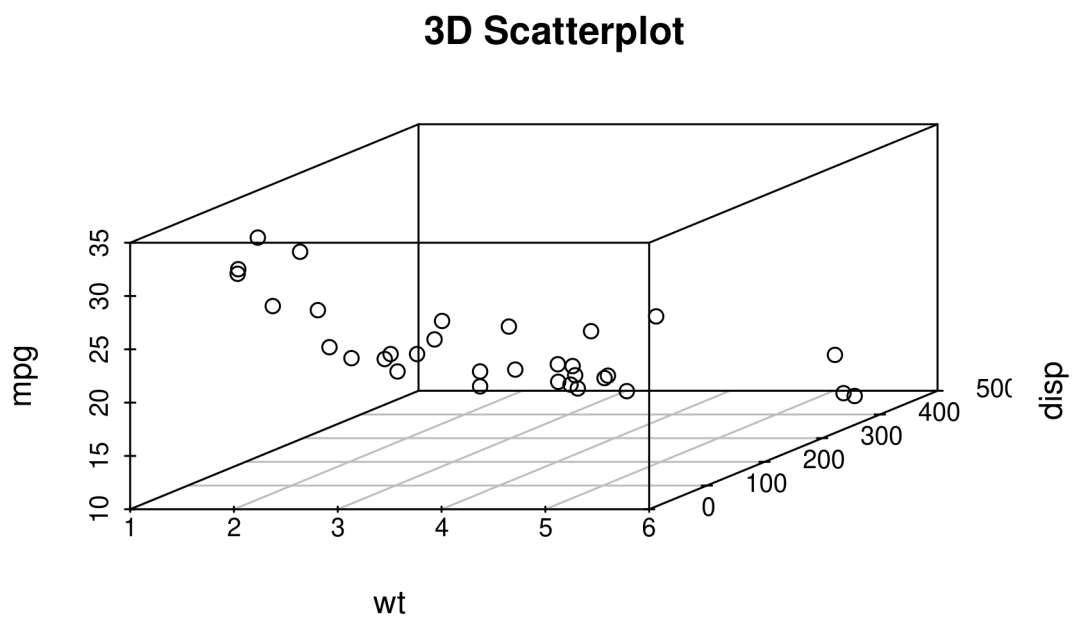



```
# 3D Scatterplot
library(scatterplot3d)
attach(mtcars)
```

5.2.5.5 There is a 3D scatterplot package

```
## The following objects are masked from mtcars (pos = 7):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt

## The following objects are masked from mtcars (pos = 10):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt

## The following object is masked from package:ggplot2:
##
##     mpg
```

```
scatterplot3d(wt,disp,mpg, main = "3D Scatterplot")
```




```
# 3D Scatterplot with Coloring and Vertical Drop Lines
library(scatterplot3d)
attach(mtcars)
```

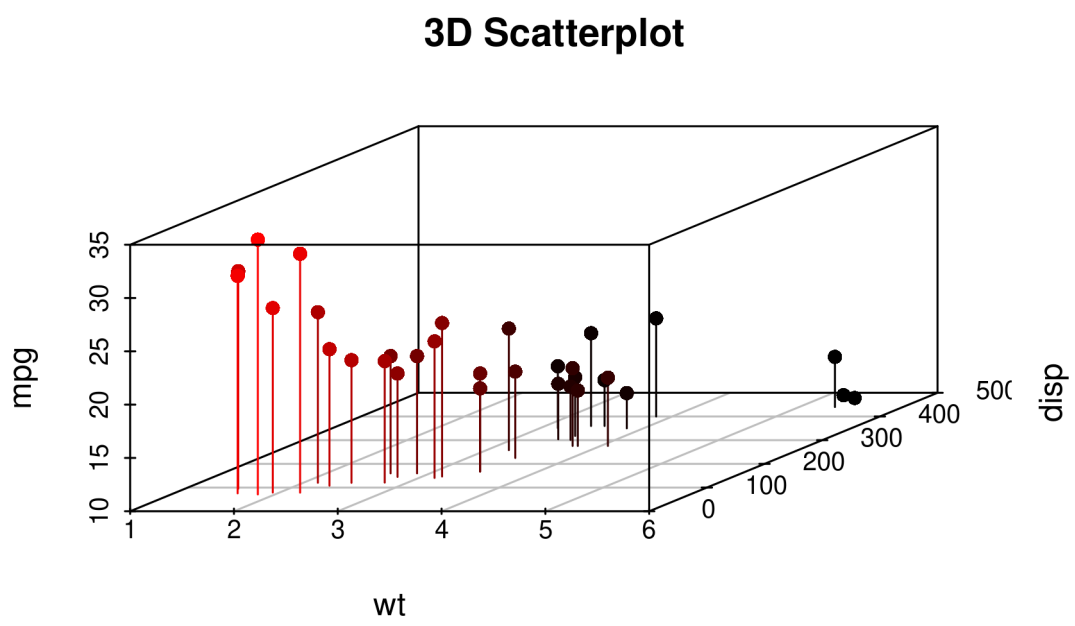
```
## The following objects are masked from mtcars (pos = 3):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following objects are masked from mtcars (pos = 8):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following objects are masked from mtcars (pos = 11):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
scatterplot3d(wt,disp,mpg, pch = 16, highlight.3d = TRUE,
              type = "h", main = "3D Scatterplot")
```




```
# 3D Scatterplot with Coloring and Vertical Lines
# and Regression Plane
library(scatterplot3d)
attach(mtcars)
```

```
## The following objects are masked from mtcars (pos = 3):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

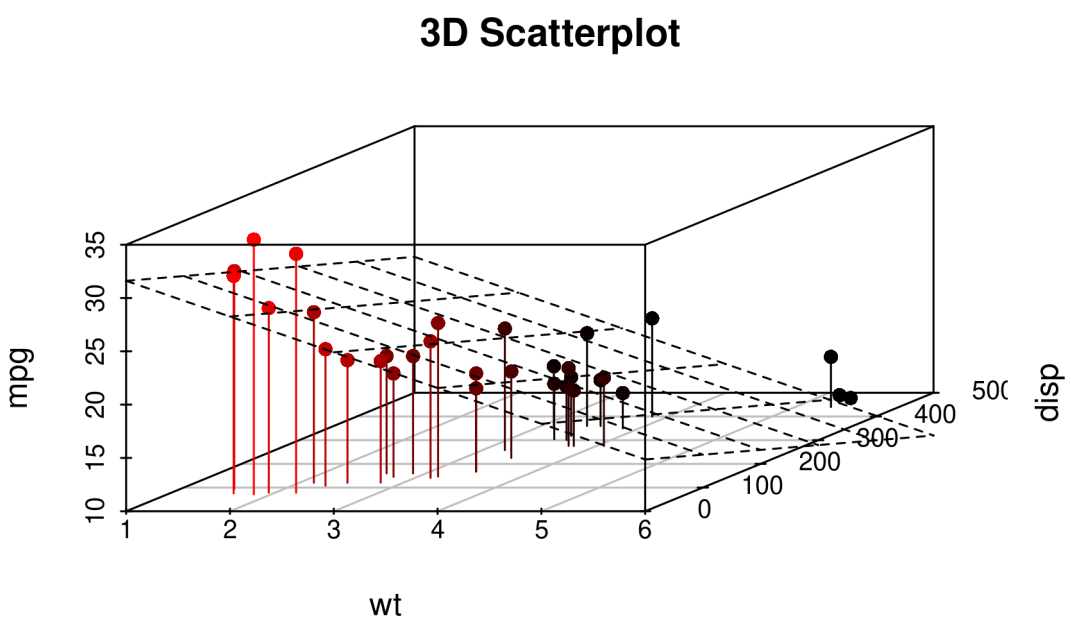
```
## The following objects are masked from mtcars (pos = 4):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following objects are masked from mtcars (pos = 9):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following objects are masked from mtcars (pos = 12):
##
##     am, carb, cyl, disp, drat, gear, hp, mpg, qsec, vs, wt
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
s3d <- scatterplot3d(wt, disp, mpg, pch = 16, highlight.3d = TRUE,
                     type = "h", main = "3D Scatterplot")
fit <- lm(mpg ~ wt+disp)
s3d$plane3d(fit)
```




```
# Spinning 3d Scatterplot
library(rgl)
```

5.2.5.5.1 3D spinning scatterplots using rgl or Rcmdr packages

```
## This build of rgl does not include OpenGL functions.  Use
##  rglwidget() to display results, e.g. via options(rgl.printRglwidget = TRUE).
```

```
plot3d(wt, disp, mpg, col = "red", size = 3)
```

5.2.6 Correlograms

Many statistical tools exist for analyzing their structure, but, surprisingly,

- there are few techniques for exploratory visual display,
  - and for depicting the patterns of relations among variables
  - in such matrices directly,
  - particularly when the number of variables is moderately large.

This describes a set of techniques we subsume under the name corrgram, based on two main schemes:

- (a) rendering the value of a correlation to depict its sign and magnitude.
  - We consider some of the properties of several iconic representations,
  - in relation to the kind of task to be performed.

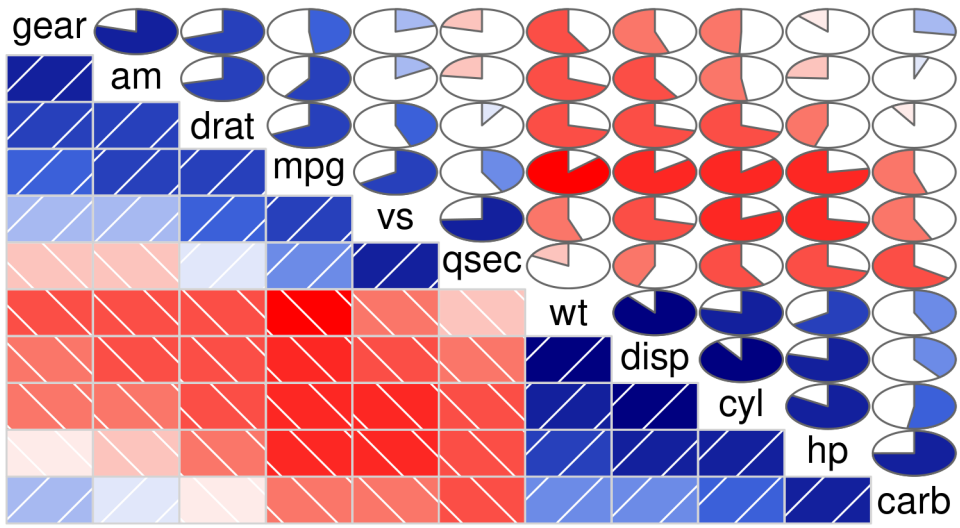
(b) re-ordering the variables in a correlation matrix - so that "similar" variables are positioned adjacently, - facilitating perception.

```
# First Correlogram Example
library(corrgram)
```

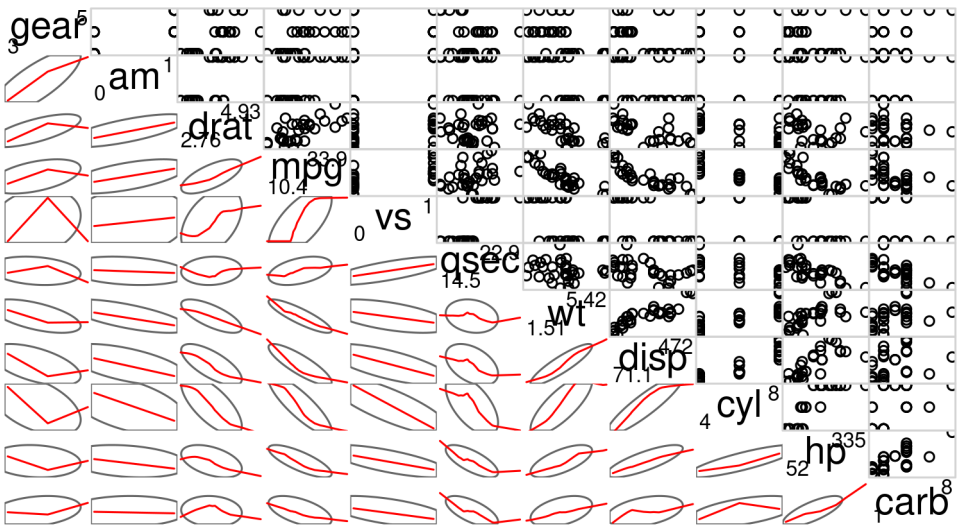
```
##
## Attaching package: 'corrgram'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     panel.cor
```

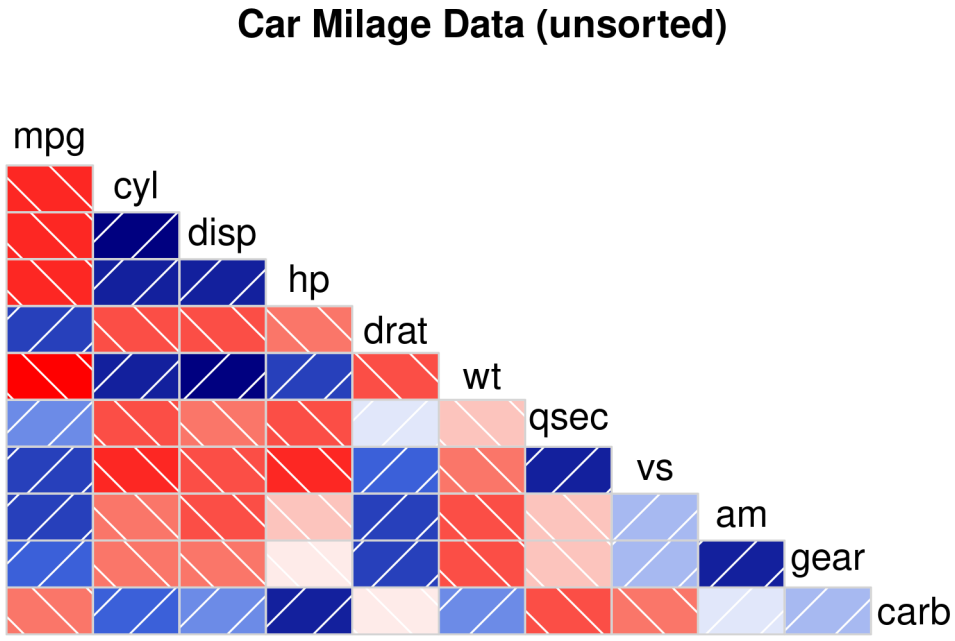
```
corrgram(mtcars, order = TRUE, lower.panel = panel.shade,
         upper.panel = panel.pie, text.panel = panel.txt,
         main = "Car Milage Data in PC2/PC1 Order")
```




```
# Second Correlogram Example
library(corrgram)
corrgram(mtcars, order = TRUE, lower.panel = panel.ellipse,
         upper.panel = panel.pts, text.panel = panel.txt,
         diag.panel = panel.minmax,
   main = "Car Milage Data in PC2/PC1 Order")
```




```
# Third Correlogram Example
library(corrgram)
corrgram(mtcars, order = NULL, lower.panel = panel.shade,
         upper.panel = NULL, text.panel = panel.txt,
         main = "Car Milage Data (unsorted)")
```



5.2.6.1 Psych package is very popular in our group

From Degradation Science COSSMS review

Fig. 9. (a) Schematic diagram of PV module and microinverter setup. (b) Comparison of actual microinverter temperature and fitted microinverter temperature for the microinverters connected to four different PV module brands during noon time on a typical cloudy day. (c) Pairs plot and correlation coefficient between different environmental and application stressors. Irradiance, wind speed and ambient temperature (Ambient.T) are the environmental stressors. PV module temperature (Module.T), PV module brand(Brand), AC power (Power) and microinverter temperature (Micro.T) are application stressors.

```
library(GGally)
library(ggplot2)
pairs(iris)
```

5.2.6.2 As is the ggpairs function of the ggally package

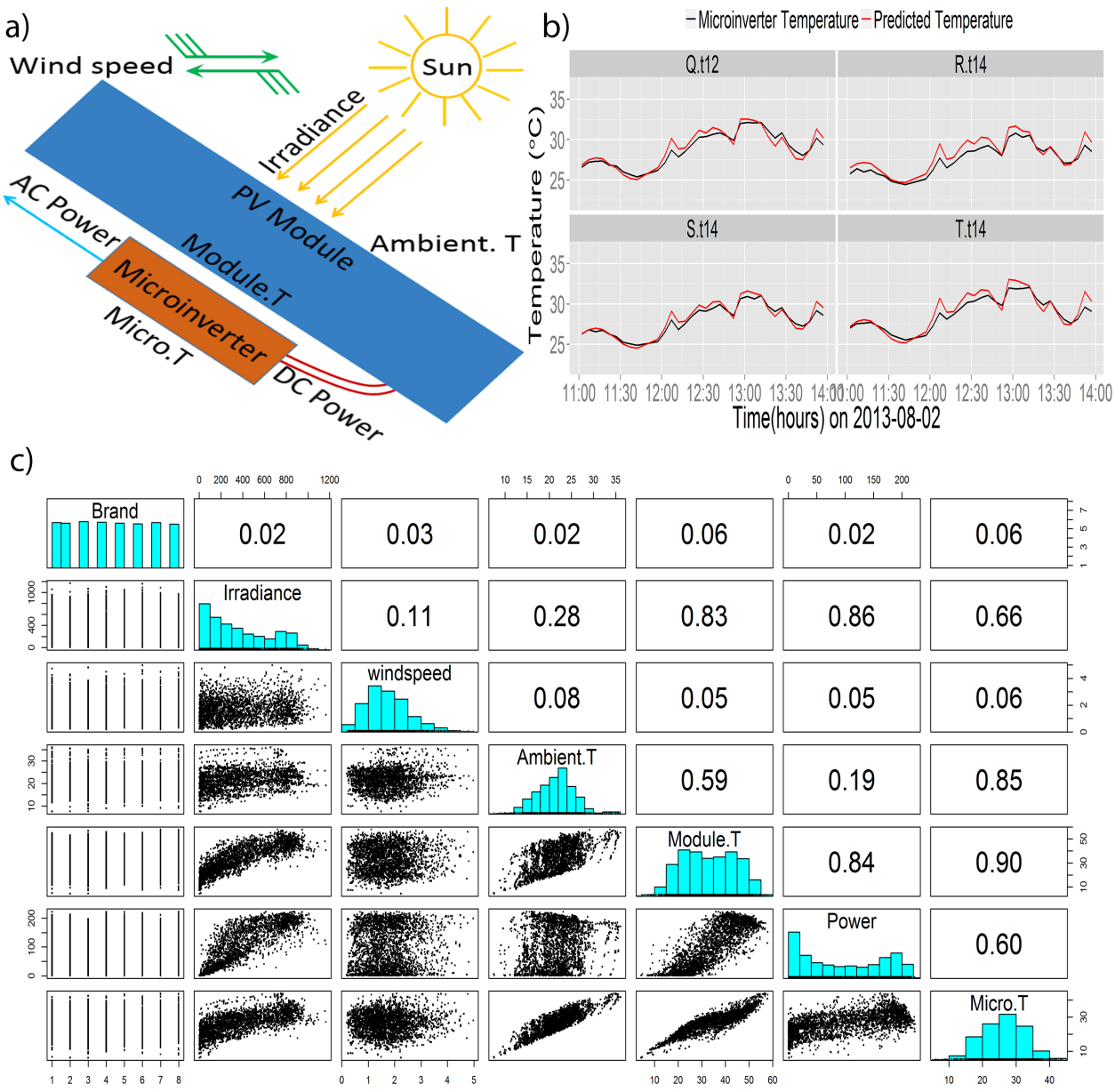
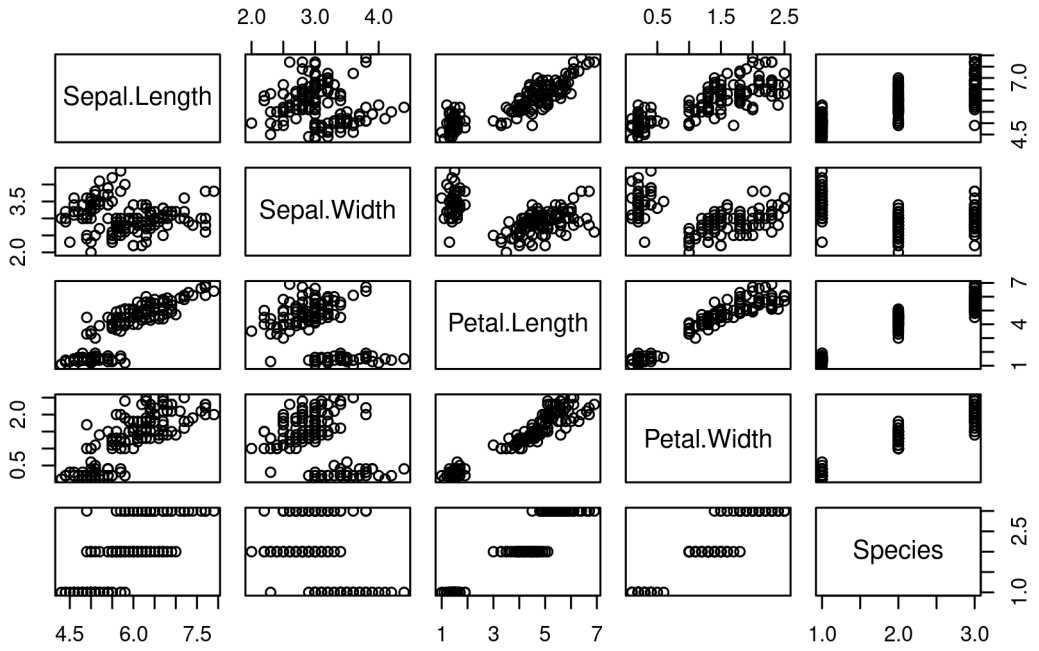
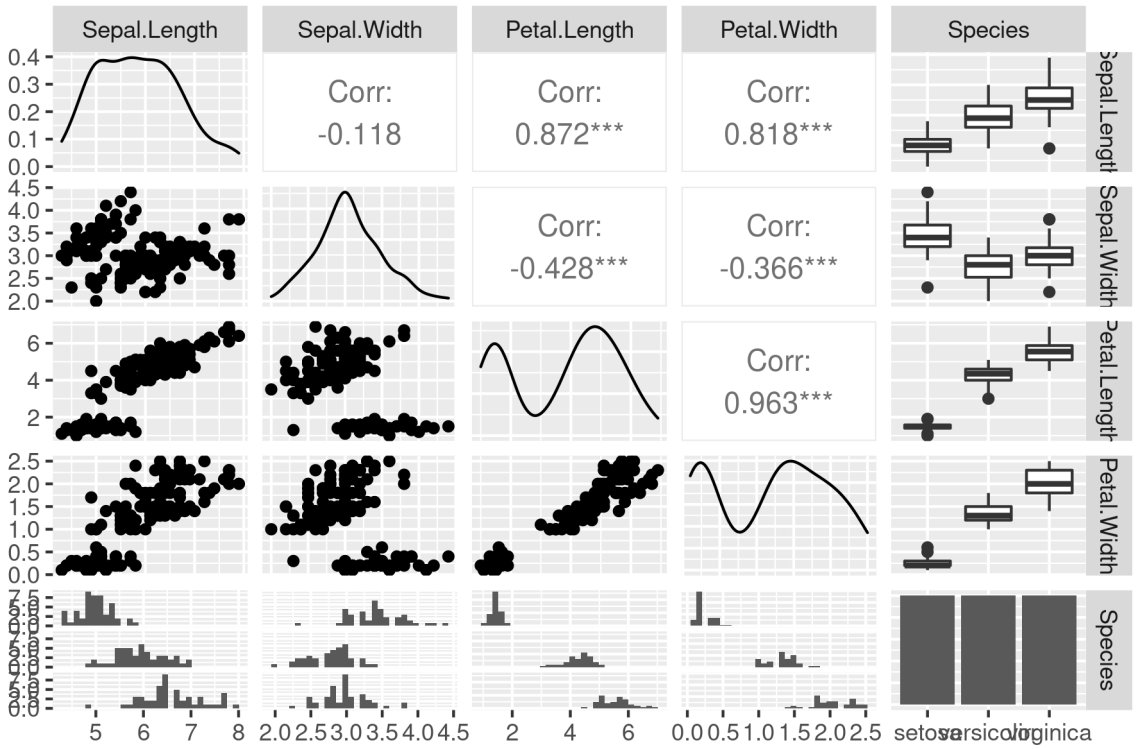


Figure 1: Figure 9.




```
ggpairs(iris)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Compared to pairs()

- we are not just limited to scatterplots,

And with ggpairs()

- we can see various plots for numeric/categorical variables.

What ggpairs provides us more than pairs()

- Along with scatterplots
  - correlation coefficients are also providing in the same panel.
- Density plots for every numeric continuous variable
  - help us to identify skewness, kurtosis and distribution information.
- Box plots are used to represent a statistical summary
  - for categorical and respective numeric variable.
- Bar charts
- Histograms

Additionaly you can also explore ggcorr() method of GGally

- which gives graphical representation of only correlation coefficients
- without any plots.

So we can conclude that the panel

- gives ample necessary insights
- but the process is a little bit time consuming
  - compared to pairs() method.

Using some level of pre-examination over your dataset

- at the primary stage
- can help us identify significant variables
  - for suitable regression models.

5.2.7 An example of EDA with pipes and pairs plots

Library in the packages we will use

```
library(ggplot2) # load package

? ggplot2

library(dplyr)  #load package
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:car':
##
##     recode
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
? dplyr
```

5.2.7.1 The Diamonds dataset

Do load the Diamonds dataset and do some quick EDA

```
data(diamonds)  # load dataset that comes with ggplot

head(diamonds) # Return the First or Last Part of an Object
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
## 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

```
str(diamonds) # Compactly Display the Structure of an Arbitrary R Object
```

```
## tibble [53,940 x 10] (S3: tbl_df/tbl/data.frame)
##  $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

```
summary(diamonds) #  produce result summaries of the results of
```

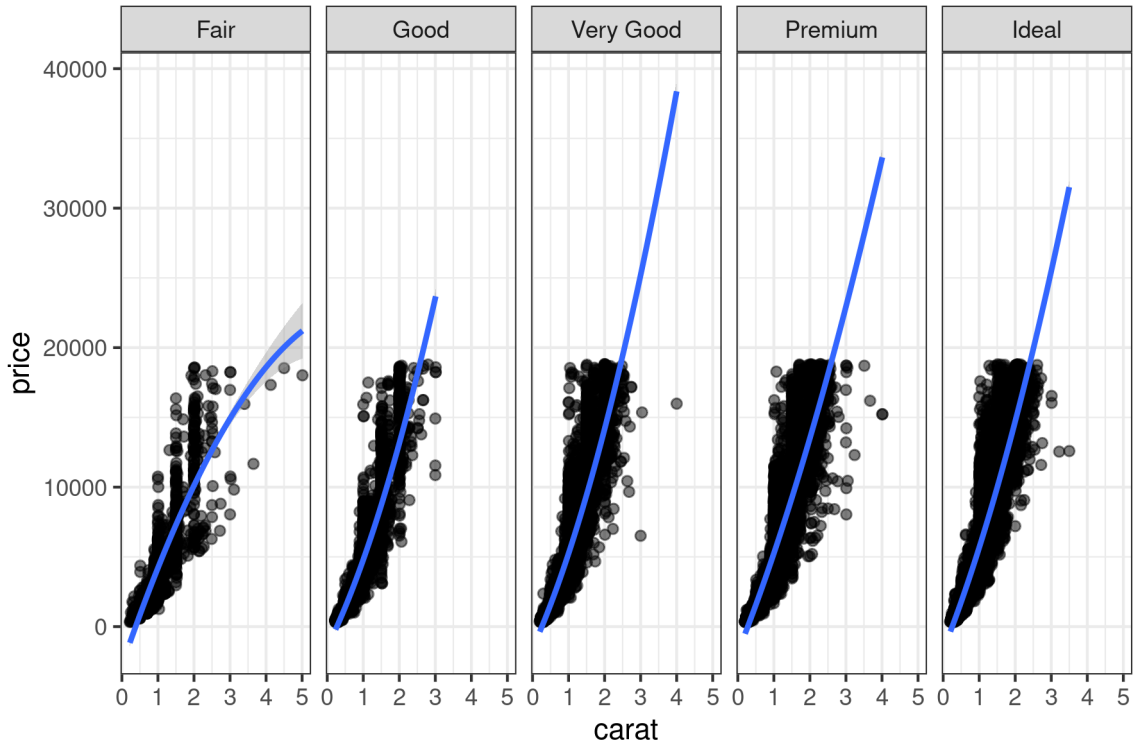
```
##      carat               cut        color        clarity          depth
##  Min.   :0.2000   Fair      : 1610   D: 6775   SI1     :13065   Min.   :43.00
##  1st Qu.:0.4000   Good      : 4906   E: 9797   VS2     :12258   1st Qu.:61.00
##  Median :0.7000   Very Good:12082   F: 9542   SI2     : 9194   Median :61.80
##  Mean   :0.7979   Premium   :13791   G:11292   VS1     : 8171   Mean   :61.75
##  3rd Qu.:1.0400   Ideal     :21551   H: 8304   VVS2    : 5066   3rd Qu.:62.50
##  Max.   :5.0100                     I: 5422   VVS1    : 3655   Max.   :79.00
##                                     J: 2808   (Other): 2531
##      table           price             x                y
##  Min.   :43.00   Min.   :  326   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:56.00   1st Qu.:  950   1st Qu.: 4.710   1st Qu.: 4.720
##  Median :57.00   Median : 2401   Median : 5.700   Median : 5.710
##  Mean   :57.46   Mean   : 3933   Mean   : 5.731   Mean   : 5.735
##  3rd Qu.:59.00   3rd Qu.: 5324   3rd Qu.: 6.540   3rd Qu.: 6.540
##  Max.   :95.00   Max.   :18823   Max.   :10.740   Max.   :58.900
##
##        z
##  Min.   : 0.000
##  1st Qu.: 2.910
##  Median : 3.530
##  Mean   : 3.539
##  3rd Qu.: 4.040
##  Max.   :31.800
##
```

```
                # various model fitting functions
```

5.2.7.2 Now some compact ggplot2 EDA code Use the pipe %>% operator

- Which simply passes the output of the left operator
- As the first argument to the right operator

```
diamonds %>%
  ggplot(aes(x = carat,y = price)) +  # aes is aesthetic mapping
    geom_point(alpha = 0.5) + # each data point as a point
    facet_grid(~cut) + # facet the scatter plot on cut, color or clarity
    stat_smooth(method = lm, formula = y ~ poly(x,2)) + # fit a 2nd order lin. model
    theme_bw()
```



With this simple visualization,

- We can quickly see that price increases with carat size,
  - The relationship is nonlinear,
- There are some outliers,
  - And the relationship does not depend too heavily on cut.

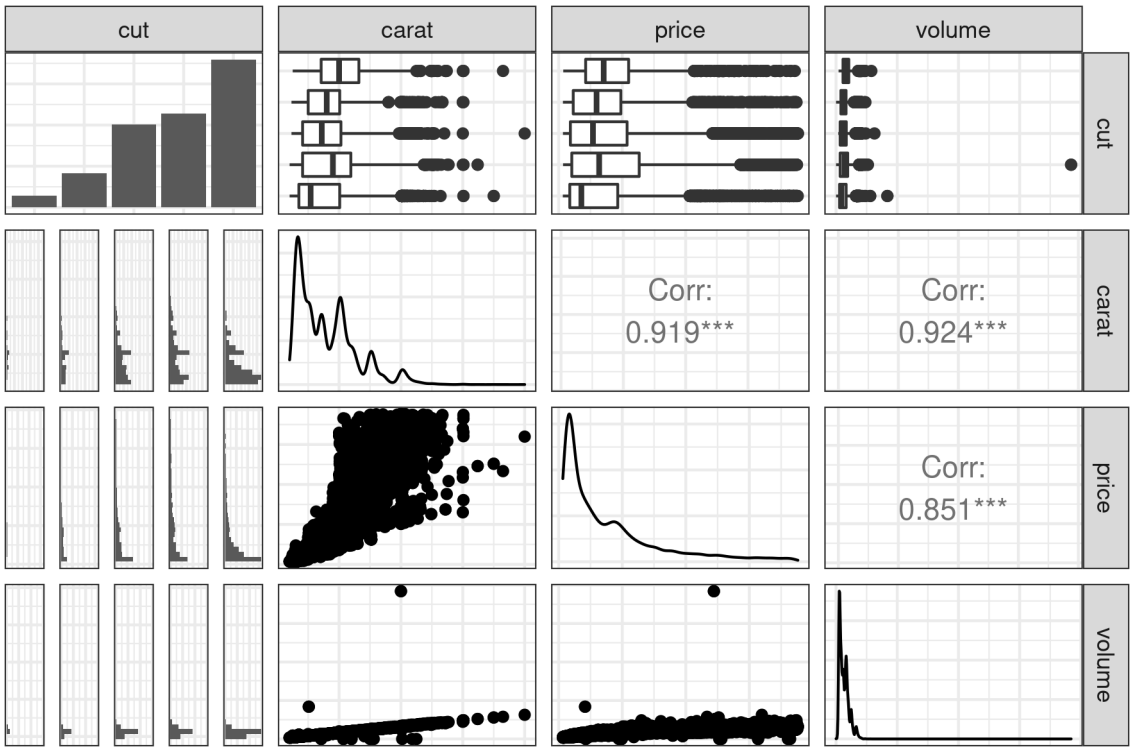
Now lets use GGally and GGpairs packages

These packages are extensions to GGplot 2

```
library(GGally) # a ggplot2 extention; gallery of plot templates

? GGally  # This doesn't work for the GGally package

?? GGally # So try this
```

```
diamonds %>%
  mutate(volume = x*y*z) %>%    # in the pipe calculate the volume
  select(cut, carat, price, volume) %>%
  sample_frac(0.5, replace = TRUE) %>%
  ggpairs(axisLabels = "none") +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



5.2.7.3 Citations

1. Scatter Plot Matrices in R
2. Simple Scatterplot
3. Correlograms + Michael Friendly Corrgrams: Exploratory displays for correlation matrices
4. Nice pairs plots with correlation coefficients in the upper quandrant + psych: Procedures for Psychological, Psychometric, and Personality Research + Using R and psych for personality and psychological research