R Commander (Rcmdr)

Dane Boring

12/6/19

What is it?

- Wikipedia: “R Commander is a GUI for the R programming language”
  - GUI – Graphical User Interface (point and click)
- rcommander.com: “R provides a powerful and comprehensive system for analysing data and when used in conjunction with the R-commander it also provides one that is easy and intuitive to use
- Dane: it’s between ease-of-use for Excel and the less intuitive coding for more individualized procedures in R

What is a GUI? (compared to R)

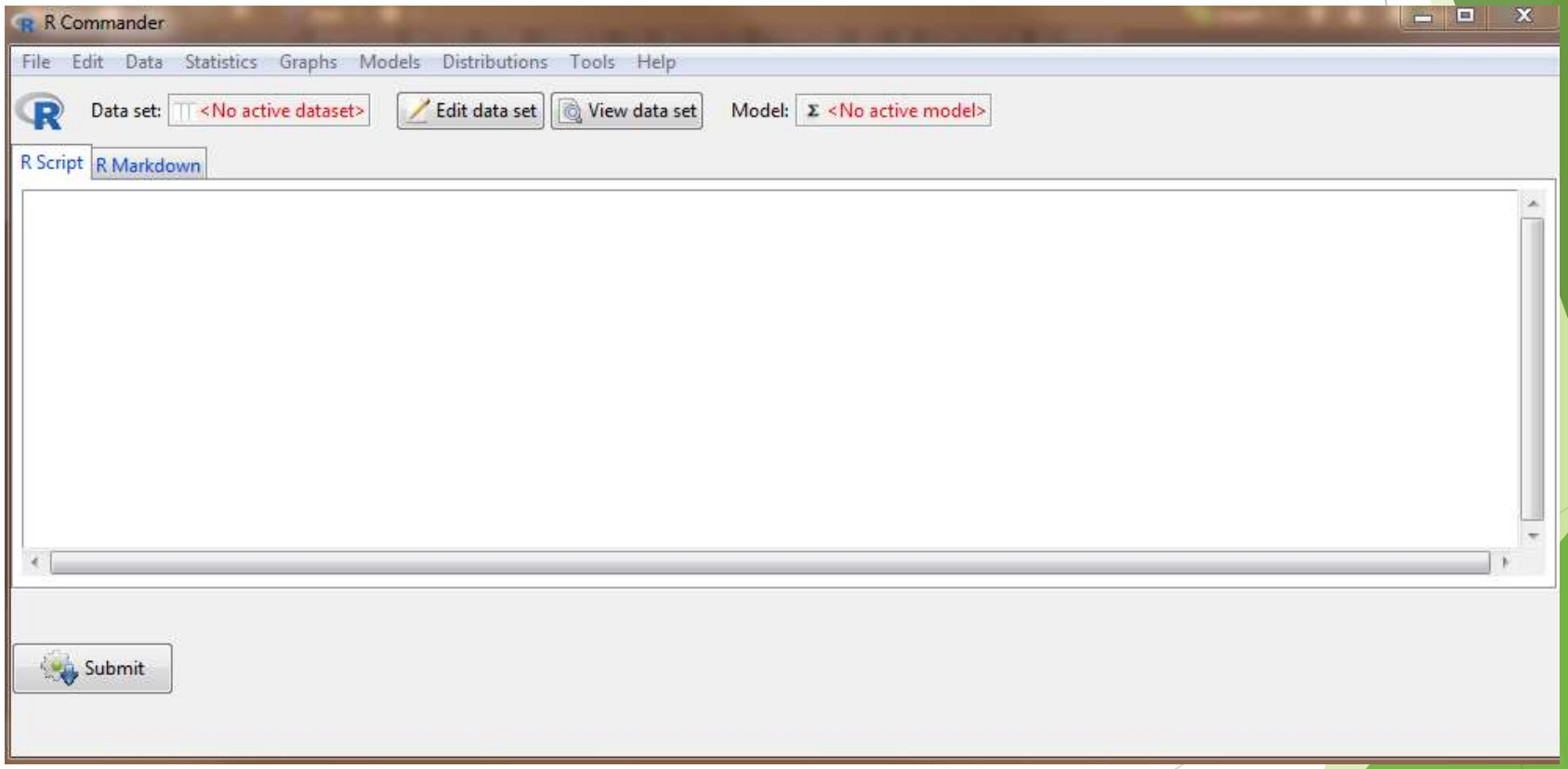
1. A computer running at its most basic is in binary (the 1's and 0's you've seen referred to in movies)
2. Nobody ($p<0.05$) actually uses this. In school, we're taught how to use the operating system (OS) Windows.
3. The OS helps the average Joe figure out how to make use of a computer. Thus, we have everything put into a point-and-click visualization (GUI)
4. Excel is the most commonly used/known way of pointing and clicking your way through data (GUI)

Why is coding (R) better than GUI (Excel)?

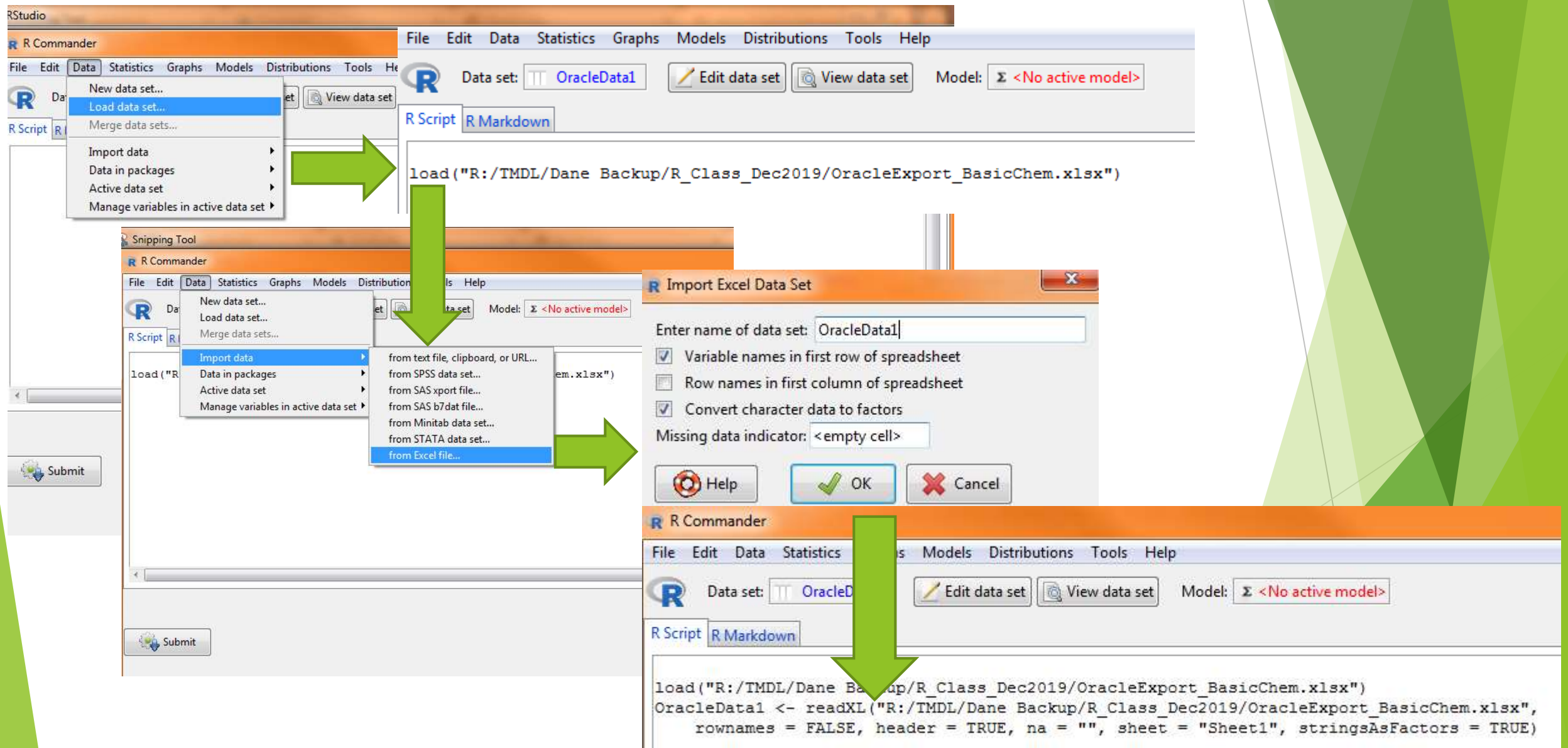
1. It's free and open source (for R)
2. It can be automated (imagine a dynamic dataset, like environmental data, that needs to be run through same steps quarterly)
3. It can be refined to every level from big to minutiae
4. Huge open source community on Github and Stack exchanges
5. GUI uses more computer resources to run since it has the graphics and buttons to process and maintain on screen

Hybrid approach? Rcmdr

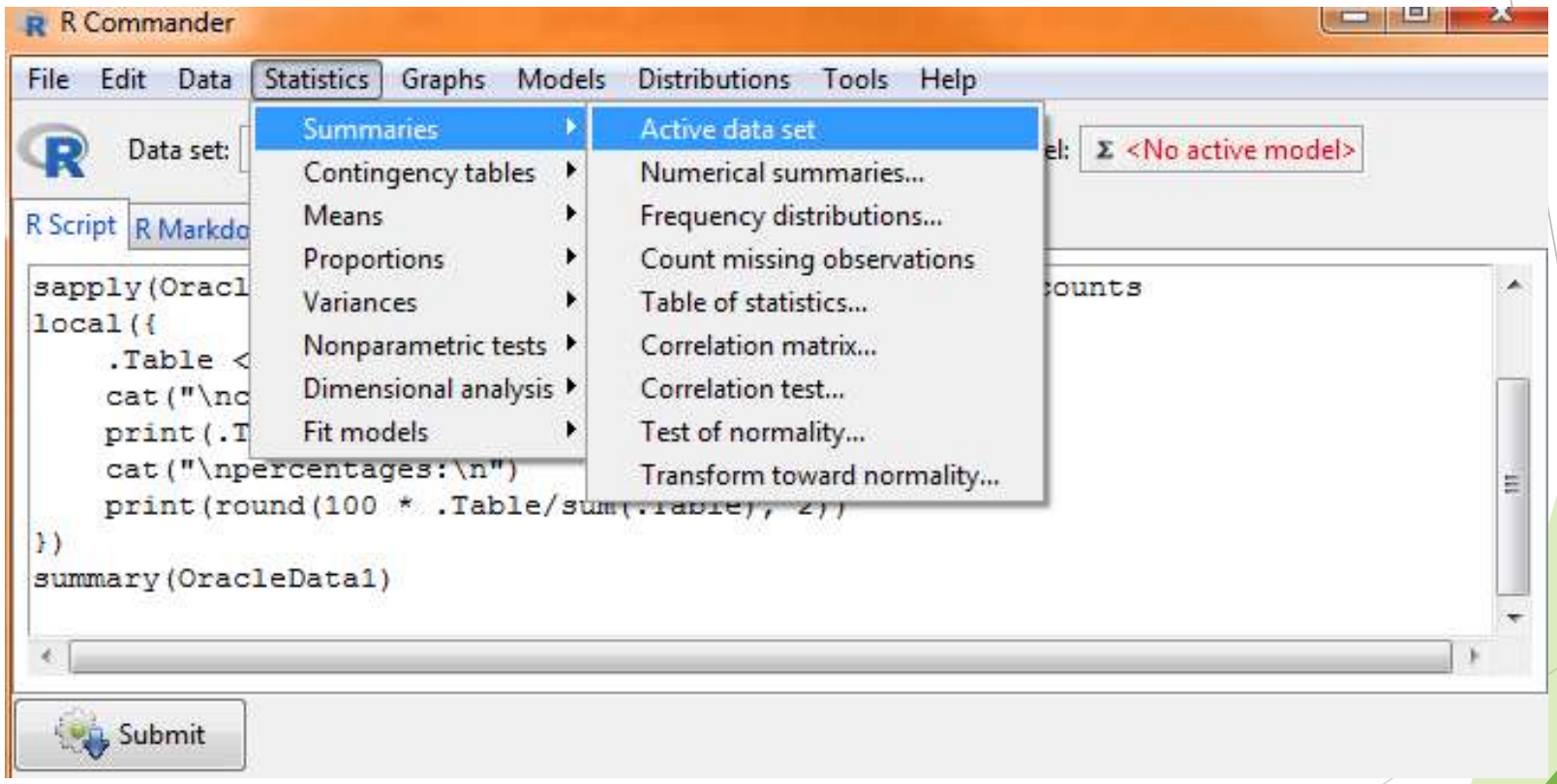
library(Rcmdr) to launch:



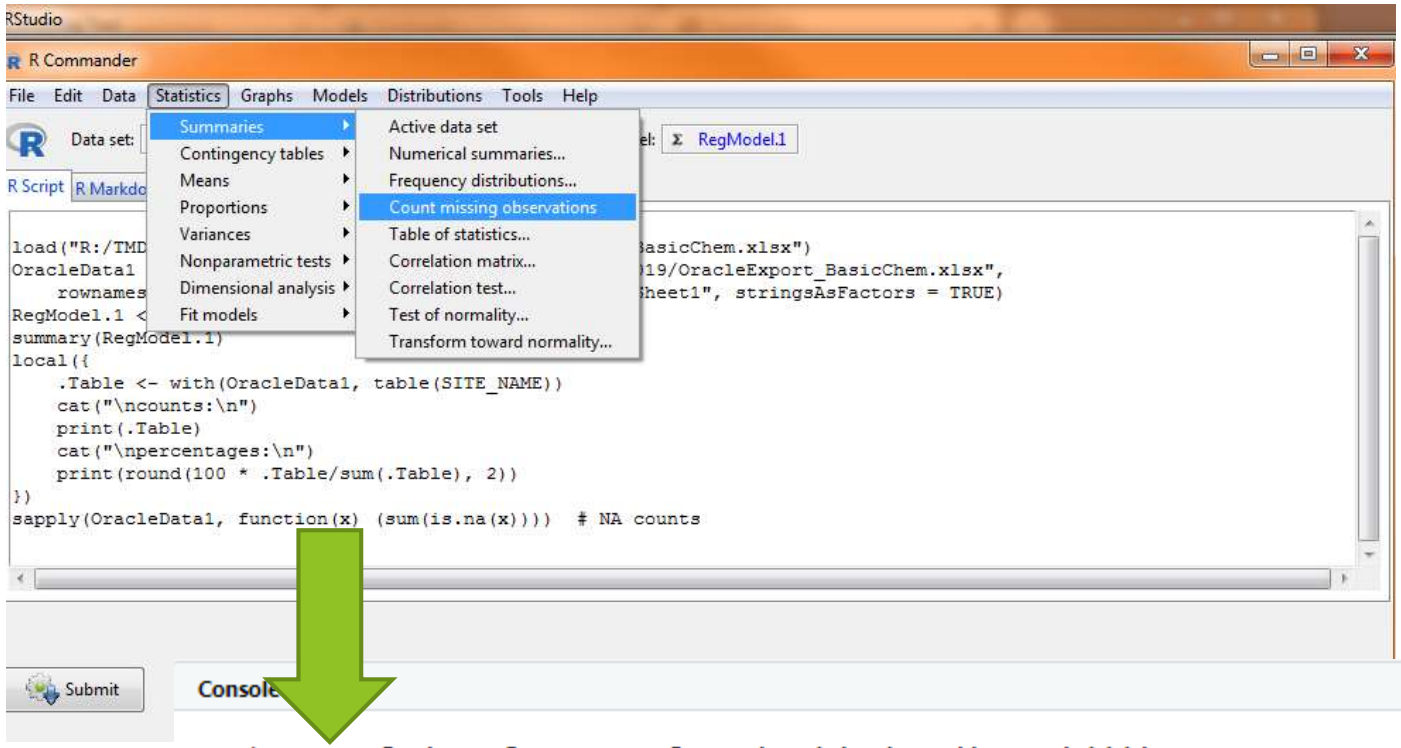
Give it a whirl



Everything at once:



Other exploratory methods:




```
Rcmdr> sapply(OracleData1, function(x) (sum(is.na(x))))  # NA counts
 PROG_CODE   SITE_NAME   SITEID_NO   SITE_CNTY        HUC8       BASIN   SITE_LOCT
         0           0       57880       57880       81938       81938       27991
 COL_DATEX   COL_DATE    COL_TIME   ALKALINTY  ALKALINTYR   CHLORIDE   CHLORIDER
         0           0           0       21344       81938       21096       81938
    DISOXY     DISOXYR     PHFIELD    PHFIELDR      SODIUM     SODIUMR   SPEC_COND
     11937       81938       37833       81938       21966       81938       19694
SPEC_CONDR   TEMP_CENT  TEMP_CENTR         TSS        TSSR   TURBIDITY  TURBIDITYR
     81938       11701       81938       24417       75612       17593       81429
```

Numerical Summaries

	mean	sd	IQR	0%
ALKALINTY	187.639273	70.0054617	93.00000	0.100
CHLORIDE	122.907490	462.2857405	84.82500	0.010
DISOXY	8.577323	3.2839337	4.10000	0.000
PHFIELD	7.825782	0.4773612	0.50000	0.000
SODIUM	93.429260	287.7716291	66.66025	0.050
SPEC_COND	965.644714	1530.6326628	659.00000	1.361
TEMP_CENT	16.668245	8.8973916	15.00000	-3.000
TSS	105.608585	421.6807284	63.00000	1.000
TURBIDITY	60.417250	267.0191075	38.00000	0.050

	25%	50%	75%	100%	n
ALKALINTY	140.00	188.000	233.00000	580.00	60594
CHLORIDE	10.51	27.200	95.33500	46857.00	60842
DISOXY	6.70	8.400	10.80000	32.40	70001
PHFIELD	7.60	7.900	8.10000	10.29	44105
SODIUM	13.90	30.007	80.56025	28370.00	59972
SPEC_COND	421.00	631.000	1080.00000	159500.00	62244
TEMP_CENT	9.00	18.000	24.00000	45.00	70237
TSS	12.00	29.000	75.00000	60700.00	57521
TURBIDITY	7.00	17.000	45.00000	32000.00	64345

	NA
ALKALINTY	21344
CHLORIDE	21096
DISOXY	11937
PHFIELD	37833
SODIUM	21966
SPEC_COND	19694
TEMP_CENT	11701
TSS	24417
TURBIDITY	17593

Table of Statistics (means)

```
Rcmdr> with(OracleData1, tapply(CHLORIDE, list(SITE_CNTY), mean, na.rm = TRUE))
       ??        AL        AN        AT        BA        BB
19.838800 25.568824  3.267500 12.754989 115.219127 14.463288
       BR        BT        BU        CA        CD        CF
15.806131 120.040041 80.824649 469.436000 209.486827 13.710432
       CK        CL        CM        CN        CQ        CR
14.858148 125.387795 373.351295 12.219506 21.195153  2.451667
       CS        CY        DC        DG        DK        DP
14.471747 52.797011 33.235833 53.166486 180.344819 10.120069
       ED        EK        EL        EW        FI        FO
74.946575  9.728571 141.389571 321.739180 151.298485 88.082333
       FR        GE        GO        GW        HG        HM
11.634856 68.104010 81.756667 30.965661 140.315000 164.413302
       HP        HV        JA        JF        JO        JW
53.218603 216.817028  8.381250 10.260538 66.634795 33.638701
       KE        KM        KW        LB        LC        LG
136.435000 201.254248 74.301282 13.230287 402.433263 54.787209
       LN        LV        LY        MC        ME        MG
11.355678 43.244904 15.078333 80.150225 1012.149205 25.016615
       MI        MN        MP        MR        MS        MT
10.290669 34.931168 20.610000  6.285677 18.218563 99.183750
       NM        NO        NS        NT        OB        OS
16.274769 13.402670 106.864578 48.407143 95.951918  8.779081
       OT        PL        PN        PR        PT        RC
258.867549 58.760775 53.712308 22.959167 66.616490 1627.876827
       RH        RL        RN        RO        RP        RS
98.239403 96.915483 473.585965 96.882308 27.725448 665.921671
       SA        SC        SD        SF        SG        SM
210.151766 24.603750 24.763750 1386.274483 192.074073 81.575152
       SN        SU        TR        WA        WB        WL
41.274019 304.141535 66.659077  3.665000  9.691000 21.645278
       WO        WS        WY
 5.521875 29.882484 62.500429
```

Kruskal-Wallis and Wilcoxon Tests:

```
Rcmdr>  kruskal.test(CHLORIDE ~ SITE_NAME, data = OracleData1)

        Kruskal-Wallis rank sum test

data:  CHLORIDE by SITE_NAME
Kruskal-Wallis chi-squared = 21109, df = 1220, p-value < 2.2e-16
```

```
Rcmdr>  with(OracleData1, wilcox.test(CHLORIDE, alternative = "two.sided", mu = 0))

        Wilcoxon signed rank test with continuity correction

data:  CHLORIDE
V = 285163021, p-value < 2.2e-16
alternative hypothesis: true location is not equal to 0
```

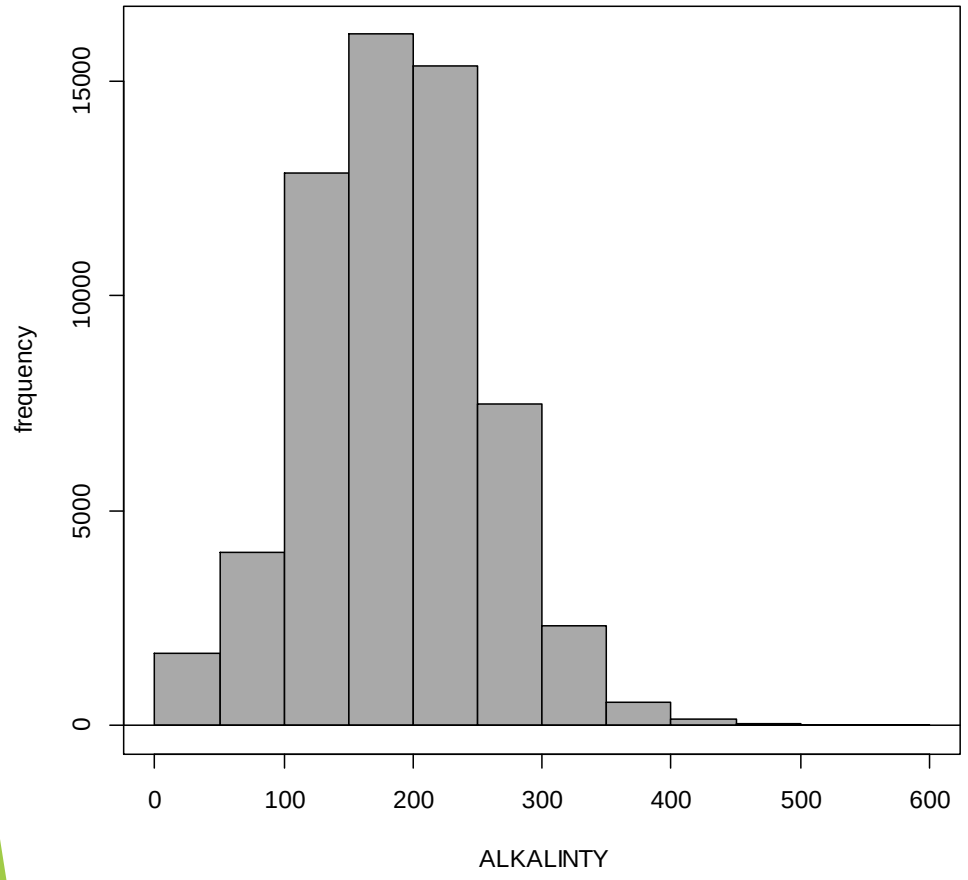
Linear Regression between Na and Cl:

```
Rcmdr>  summary(RegModel.1)

Call:
lm(formula = CHLORIDE ~ SODIUM, data = OracleData1)

Residuals:
    Min      1Q  Median      3Q     Max
-931.26   -5.29    7.14   13.55 1162.39

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.063465   0.449954  -37.92   <2e-16 ***
SODIUM        1.498474   0.001966  762.01   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 63.62 on 23836 degrees of freedom
  (6115 observations deleted due to missingness)
Multiple R-squared:  0.9606,    Adjusted R-squared:  0.9606
F-statistic: 5.807e+05 on 1 and 23836 DF,  p-value: < 2.2e-16
```

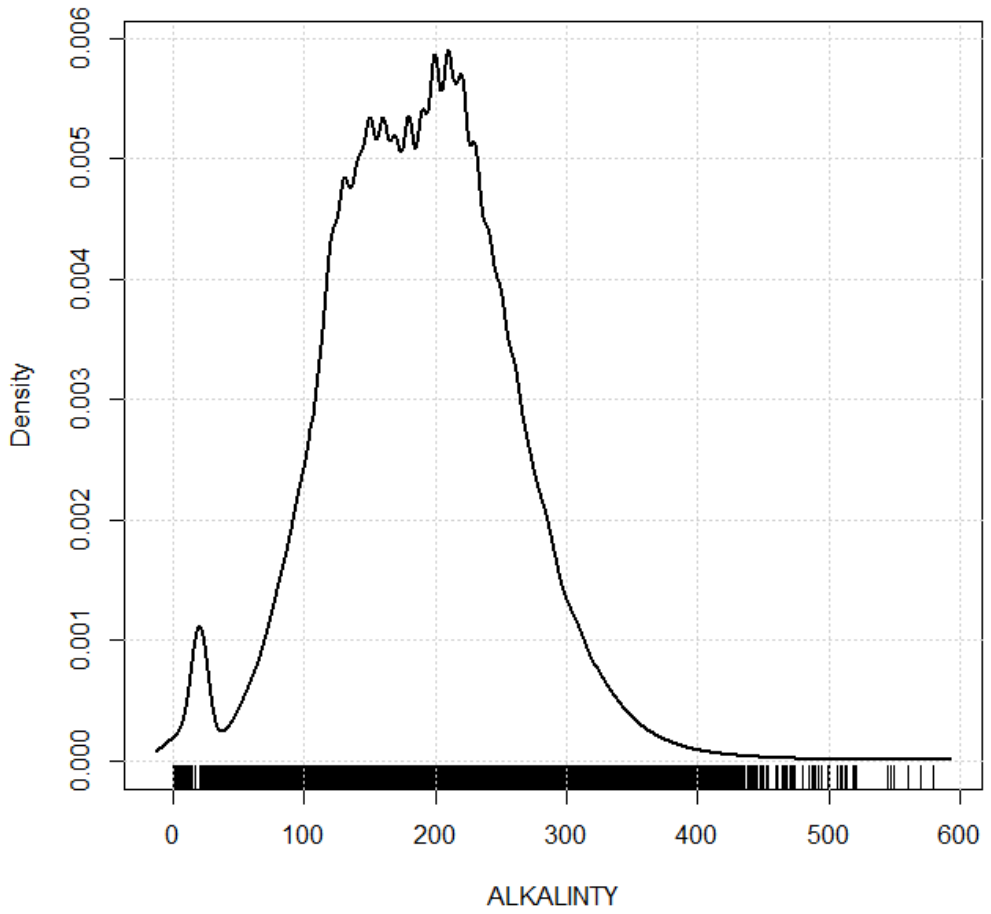
Linear Regression with multiple explanatory variables:

```
Rcmdr>  summary(RegModel.2)

Call:
lm(formula = CHLORIDE ~ SODIUM + SPEC_COND, data = OracleData1)

Residuals:
    Min      1Q  Median      3Q     Max
-940.71   -5.35    1.69    9.58 1254.74

Coefficients:
             Estimate Std. Error t value      Pr(>|t|)
(Intercept)  4.490948   0.747550   6.008 0.00000000191 ***
SODIUM       1.730032   0.006780 255.152       < 2e-16 ***
SPEC_COND   -0.044469   0.001249 -35.603       < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 61.99 on 23834 degrees of freedom
  (6116 observations deleted due to missingness)
Multiple R-squared:  0.9626,    Adjusted R-squared:  0.9626
F-statistic: 3.064e+05 on 2 and 23834 DF,  p-value: < 2.2e-16
```

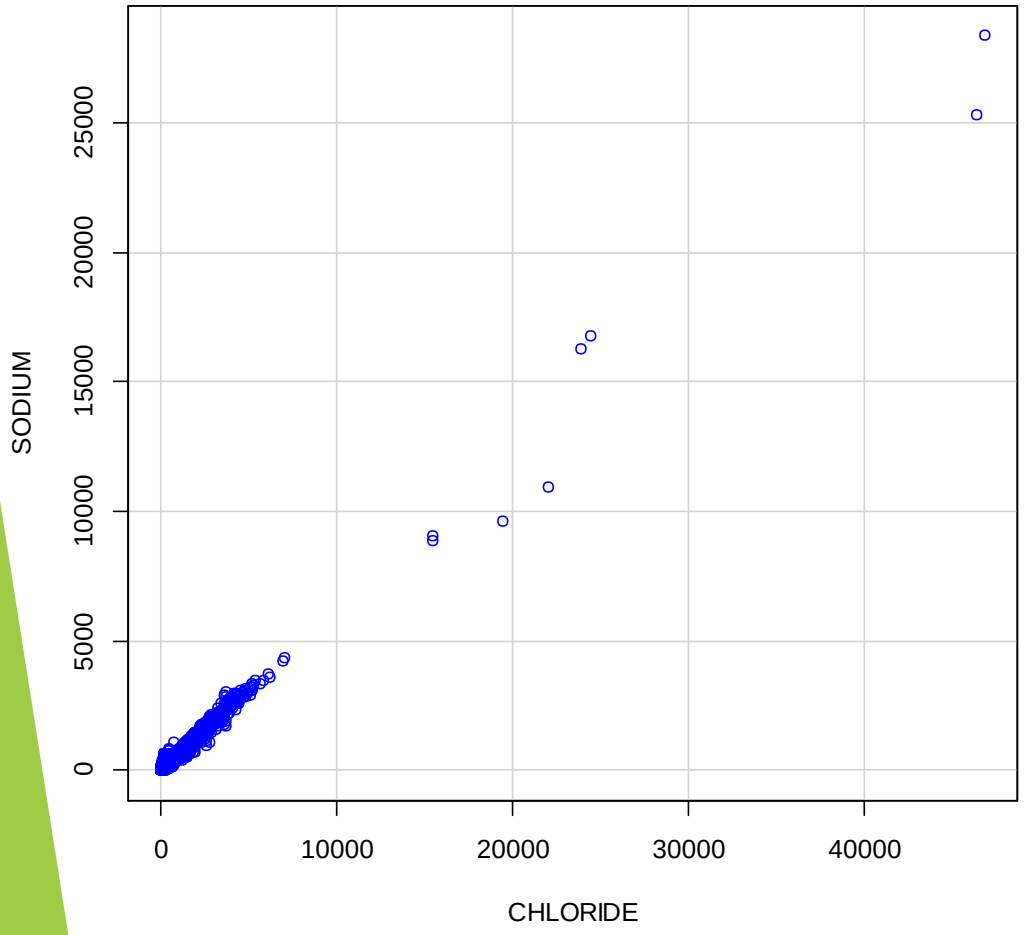
Histogram



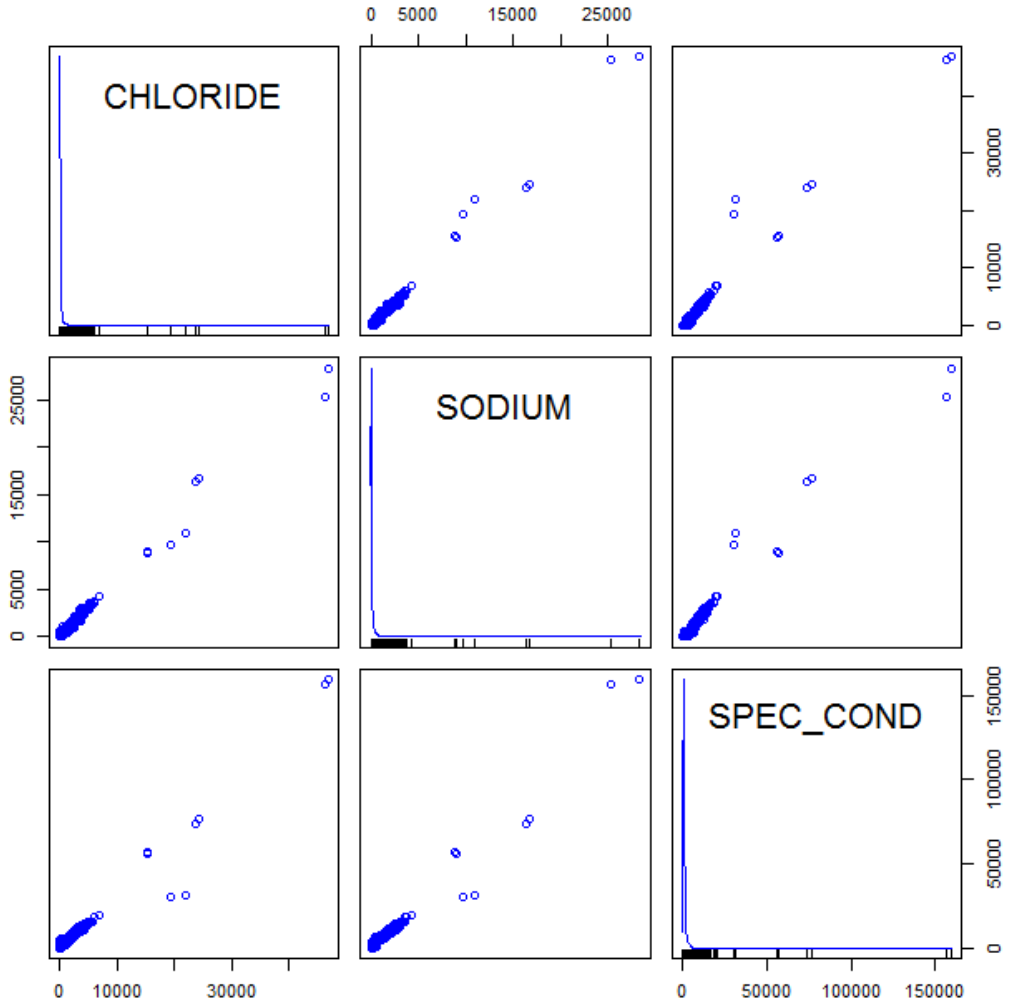
Density estimate



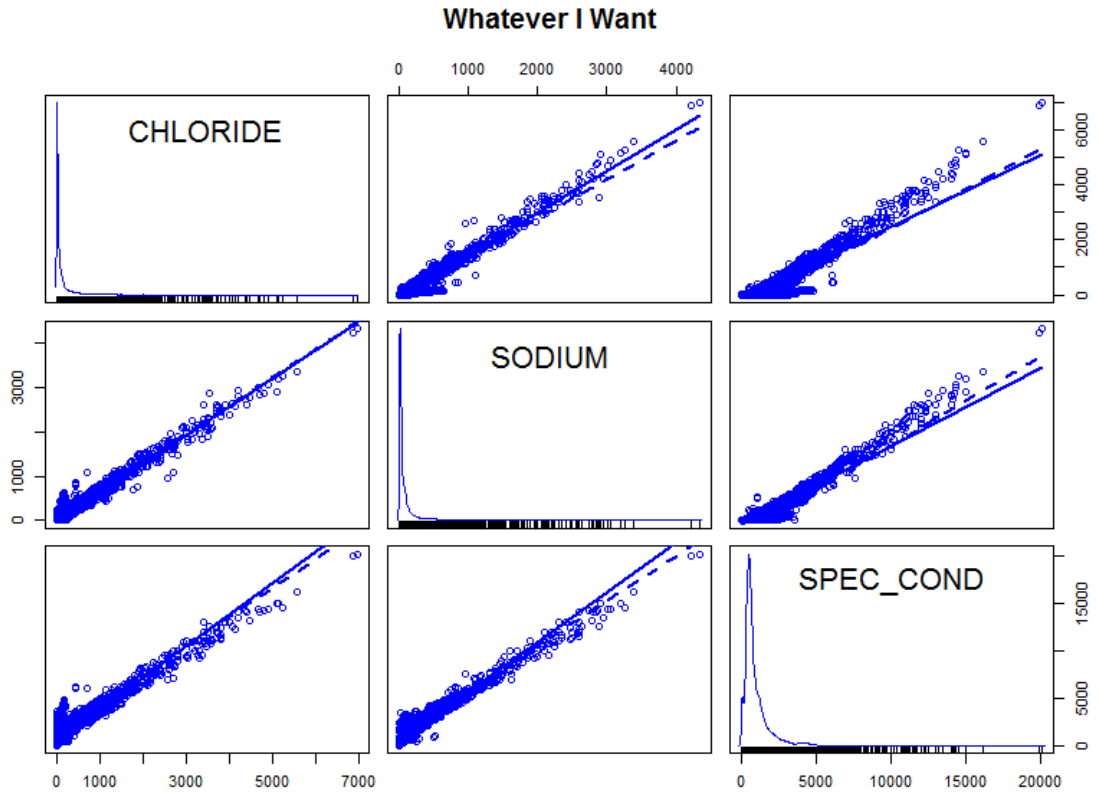
Scatterplot



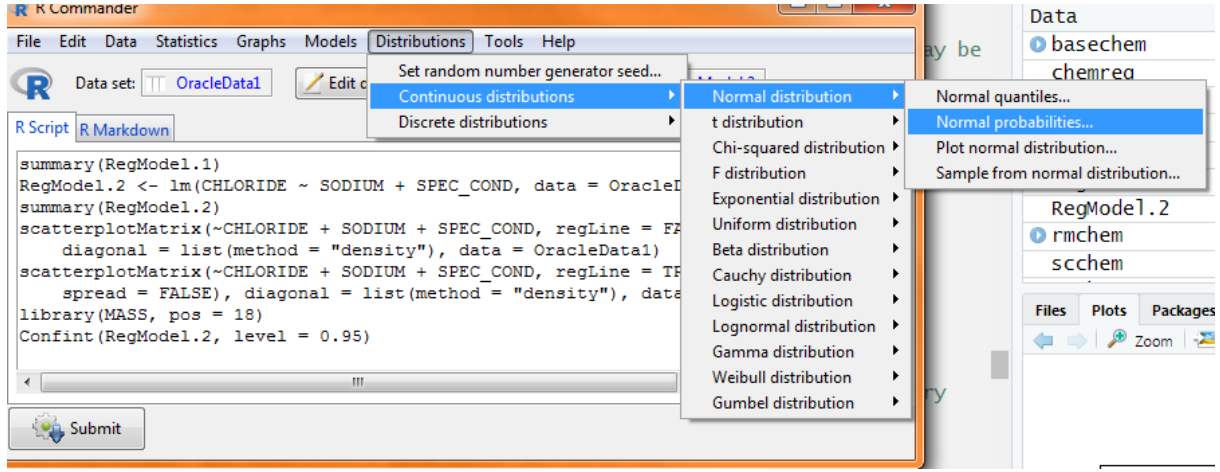
Scatterplot matrix



How about a little dressier?
Options tab!



What dat do?



Set random number generator seed:

-Computers are not random. Basically, you can introduce your own bit of randomness setting that number how you like.

Computer generated distributions are a branch of statistics I won't try to explain for lack of depth on my part, but just know these are here.

Seems great; what's the snag?

- The dataset we loaded is raw
  - What we put in has to be a finished product ready for the statistical analysis
- Not much variety involved
  - R coding allows near infinite adjustments. We've only added a title and couple lines to scatterplot matrix. How about subtitle? A fleet of these matrices for every HUC8? 'Chloride' or 'Cl' instead of "CHLORIDE" for titles? Choosing tick marks on the axes? Various color adjustments by grouping? Space of axis labels from the axis marks? Presence of axes lines even at all?
  - We're only using the basic plotting function here. No interactive plots are going to come out of this, we'll be missing certain statistical packages not Rcmdr friendly, and besides being free we're not making much more use of this over Excel.
- No automation!

Why Rcmdr if it lacks the full suite of R capability?

- Easier getting into a hot tub slowly than jumping in head-first
  - Coding is different than how we're taught to use computers in school. You actually have to understand the workings of it to use it best. Going straight from point-and-click to coding is not intuitive and takes time getting used to.
- You can click the basic solution into existence there and modify the code for your specifics

```
scatterplotMatrix(~CHLORIDE + SODIUM +
    spread = FALSE), diagonal = list(m
```

  - Copy and paste to your script then use ?function() to get a start on the arguments to make it your own
- It's a good easy starting point, especially for non-common everyday type users having to remember the coding
  - Again, it's a use-it or lose-it method. Coding takes practice and critical thinking, and Rcmdr helps with if you've been away from it for a while and lose-it per se.

Dane's summary

- Rcmdr is a useful tool for quick answers to some questions, much like Excel would provide. Additionally, it provides the code you can copy for your own use.
- However, it undercuts the usefulness of writing the code vs a GUI approach: OPTIONS
  - Handling datasets (detection limits, weird stuff, corrections, etc.)
    - Polished dataset needed in Rcmdr
  - No automation
    - Running loops to mass produce plots, results, etc.
  - Slower to set up for running after first time