The Briefest R overview, Ever

Paul Johnson and Ben Kite[1]

[1]Center for Research Methods and Data Analysis

2017

Outline

This is Brief

- This talk introduces the vital R terminology and usage necessary to get started with structural equation modeling
- Proficient use of R will require
  - practice,
  - a lot more reading, and
  - possibly some additional workshops at a University near you.

This is an Sweaved Document

- The R code displayed below is embedded within the presentation file
- When the PDF is produced, we also export the R code.
- We "zip" together the folder structure and distribute everything, data and source files included.
- You are free to explore these files, try to run the R commands for yourself.
- But I have to warn you about something
- **CAUTION: EXTRACT** the zipped folder. Drag and drop the folder out of the zip file. In Win10, one can be fooled because the file manager has an Archive Inspector mode that looks similar, but it is not extracted.

After Extracting, look around

- Use the file manager, look for a directory "presentation".
- The PDF should be in there, "sem-2-1-R.pdf".
- You should also see other files we use to produce this document.
- Look for a file named "sem-2-1-R.R"
- Use "Open With", choose RStudio, Emacs, Notepad++, R.app, or whatever R editor you use.
- You can follow along, run the "chunks". But be warned
    - my first 2 chunks have some boilerplate that you should not run because they change your session options
    - **This will fail if you don't use "Open with"** to launch R. I'll explain why you should NOT launch R (or RStudio) from the application icon. Do use "Open with".

All data analysis consistats of 6 steps

1. Data import
2. Recoding
3. Exploration
4. Analysis
5. Export of Tables & Graphs
6. Writeup

Data Input Formats

- Base R includes importers for input data files that are saved in 2 formats
  1. R Serialization Data structures ("rds" files)
  2. text files ("csv" comma separated, but also "tab" or other separators).
- R packages can be loaded which are able to import files from
  - SPSS, Stata, SAS
  - Excel
  - SQL
  - Others

Check which data files we have in "../data"

- R can interact with your operating system,

```
list.files("../data")
```

```
[1] "affect.csv"
```

look in neighboring folder "../data"

- R can create directories (`dir.create()`), copy files (`file.copy()`), and do just about anything else we want to do (e.g., delete files).

Use read.table to import the csv file

```
affect <- read.table("../data/affect.csv", header
    = TRUE, sep = ",", stringsAsFactors = FALSE)
```

- First argument is a file name (not necessary to name arguments)
- 3 named arguments:
  - `header = TRUE` : use the first row as variable names
  - `sep = ","` : use the comma as the separator
  - `stringsAsFactors = FALSE` : I will make factors myself, I don't want R to do it for me

Check the result

- That thing is a data.frame object

```
str(affect)
```

```
'data.frame': 380 obs. of 19 variables:
 $ Agency1  : num 3.5 2.5 1.83 2.77 3.17 ...
 $ Agency2  : num 4 3.17 2 3.06 3.33 ...
 $ Agency3  : num 4 3 1.5 2.36 2.83 ...
 $ Intrin1  : num 4 3.21 3 3.13 3.5 ...
 $ Intrin2  : num 4 2 3 4 4 2.5 3.5 3 2 3.5 ...
 $ Intrin3  : num 4 3 2 3 4 3 4 2 3 3 ...
 $ Extrin1  : num 1 1.83 1 1.08 1.83 ...
 $ Extrin2  : num 1 2.67 1 1.17 2 ...
 $ Extrin3  : num 1.5 1.83 1 1 1.83 ...
 $ PosAFF1  : num 4 3 3.02 3 3.78 ...
 $ PosAFF2  : num 4 3.5 2.5 2.5 3.5 3 2.5 2 3 3 ...
 $ PosAFF3  : num 4 2.5 3 3 3 3 3 2.5 3.5 3 ...
 $ NegAFF1  : num 1 1.5 1 2.5 2.5 2 1 1 2 2.5 ...
 $ NegAFF2  : num 1 1.69 1 2.5 2 ...
 $ NegAFF3  : num 1 1.5 1 1.5 3 ...
 $ Sex      : int 1 1 1 1 1 1 1 1 1 1 ...
 $ gender   : chr "male" "male" "male" "male" ...
 $ ethnicity: chr "Hispanic" "White" "White" "White" ...
```

Check the result ...

```
$ race          : chr  "Nonwhite" "White" "White" "White" ...
```

- data.frame: columns can be different types of variables
  - character: character strings
  - integer: only integers, no floating point
  - numeric: floating point
- Other types we don't see here
  - logical: Coded either `TRUE` or `FALSE`, symbols that are interpreted as 1 and 0
  - factor: R's way of creating categorical variables, either nominal or ordered
  - Date: Can subtract dates to find time between
- The same information can be encoded in different ways
  - Sex is an integer
  - Gender is a character variable
- Can see in spreadsheet-like thing with the `View()` function:

Check the result ...

```
View(affect)
```

R factor

- Convert a string to a factor

```
affect$genderf <- factor(affect$gender, levels =
    c("male", "female"))
```

- Key elements
  - The `$` symbol names a new column inside `affect` (there are several other ways to do this)
  - The function `factor()` is used to create the new variable, which is a factor
  - I chose levels in order, on purpose, to demonstrate that I can do that non-alphabetically
- Check that `gender` and `genderf` are different things

```
str(affect)
```

R factor ...

```
'data.frame': 380 obs. of  20 variables:
 $ Agency1  : num  3.5 2.5 1.83 2.77 3.17 ...
 $ Agency2  : num  4 3.17 2 3.06 3.33 ...
 $ Agency3  : num  4 3 1.5 2.36 2.83 ...
 $ Intrin1  : num  4 3.21 3 3.13 3.5 ...
 $ Intrin2  : num  4 2 3 4 4 2.5 3.5 3 2 3.5 ...
 $ Intrin3  : num  4 3 2 3 4 3 4 2 3 3 ...
 $ Extrin1  : num  1 1.83 1 1.08 1.83 ...
 $ Extrin2  : num  1 2.67 1 1.17 2 ...
 $ Extrin3  : num  1.5 1.83 1 1 1.83 ...
 $ PosAFF1  : num  4 3 3.02 3 3.78 ...
 $ PosAFF2  : num  4 3.5 2.5 2.5 3.5 3 2.5 2 3 3 ...
 $ PosAFF3  : num  4 2.5 3 3 3 3 3 2.5 3.5 3 ...
 $ NegAFF1  : num  1 1.5 1 2.5 2.5 2 1 1 2 2.5 ...
 $ NegAFF2  : num  1 1.69 1 2.5 2 ...
 $ NegAFF3  : num  1 1.5 1 1.5 3 ...
 $ Sex      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ gender   : chr  "male" "male" "male" "male" ...
 $ ethnicity: chr  "Hispanic" "White" "White" "White" ...
 $ race     : chr  "Nonwhite" "White" "White" "White" ...
 $ genderf  : Factor w/ 2 levels "male","female": 1 1 1 1 1 1 1 1 1
     1 ...
```

R factor ...

- The table function can create a quick cross-tabulation:

```
table("genderf" = affect$genderf, "gender" =
     affect$gender)
```

```
          gender
genderf   female male
  male         0  195
  female     185    0
```

The “argument names” are used as labels. They are not required:

```
table(affect$genderf, affect$gender)
```

```
          female male
  male         0  195
  female     185    0
```

- I think it is better to have a labeled variable than a 1-2 coded Sex

R factor ...

```
table("gender as factor" = affect$genderf, "Sex
    as an integer" = affect$Sex)
```

```
                  Sex as an integer
gender as factor    1   2
          male    195   0
          female    0 185
```

- I'll also need an ethnicity factor variable in a later section:

  ```
  affect$ethnicityf <- factor(affect$ethnicity)
  ```

  I allowed R to create the levels in alphabetical order, as we see here:

```
table("ethnicity factor" = affect$ethnicityf,
    "ethnicity" = affect$ethnicity)
```

R factor ...

```
                      ethnicity
ethnicity factor  Asian  Black  Hispanic  White
          Asian     38      0         0      0
          Black      0     19         0      0
          Hispanic   0      0        67      0
          White      0      0         0    256
```

- Worth mentioning: R is case sensitive
  - If I create variables, they always start with small letters
  - This input data used capital and small letters, regrettably.

About packages

- R is a computational engine
  - to which packages are attached
- The R distribution includes
  - 15 base packages (incl. base, datasets, stats, stats4, graphics)
  - 15 recommended packages (incl. foreign, MASS, mgcv, nlme, survival)
- The Comprehensive R Archive Network (CRAN) has 10,600 other "contributed" packages.
- In the base collection, we have functions to conduct analysis that was considered vital in 2000, such as regression, generalized linear models.
- The R distribution includes many famous graph-making functions, along with an underlying structure which is built-upon by many other graphics packages.

About packages ...

- Within an R program, functions like `install.packages()` , `update.packages()` , and `library()` are used to interact with the package system
- Over the years, R designers have added functionality for package management, especially the ability to keep installed packages in separate folders and to allow non-administrator users to install their "own" packages separate from the ones managed by the administrator.

Data Descriptions

```
summary(affect)
```

```
    Agency1          Agency2          Agency3          Intrin1
 Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
 1st Qu.:2.052   1st Qu.:2.167   1st Qu.:2.167   1st Qu.:2.500
 Median :2.494   Median :2.500   Median :2.500   Median :3.000
 Mean   :2.442   Mean   :2.550   Mean   :2.544   Mean   :3.002
 3rd Qu.:2.833   3rd Qu.:2.898   3rd Qu.:2.833   3rd Qu.:3.500
 Max.   :4.000   Max.   :4.000   Max.   :4.000   Max.   :4.000
    Intrin2          Intrin3          Extrin1
 Min.   :1.000   Min.   :1.000   Min.   :0.9717
 1st Qu.:2.500   1st Qu.:2.500   1st Qu.:1.3190
 Median :3.000   Median :3.000   Median :1.5000
 Mean   :2.987   Mean   :3.080   Mean   :1.6151
 3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:1.8333
 Max.   :4.025   Max.   :4.077   Max.   :3.5215
    Extrin2          Extrin3           PosAFF1
 Min.   :1.000   Min.   :0.9548   Min.   :1.000
 1st Qu.:1.185   1st Qu.:1.1667   1st Qu.:2.744
 Median :1.538   Median :1.5000   Median :3.023
 Mean   :1.686   Mean   :1.6333   Mean   :3.136
 3rd Qu.:2.000   3rd Qu.:1.9320   3rd Qu.:3.500
 Max.   :3.500   Max.   :3.8397   Max.   :4.000
```

Data Descriptions ...

```
    PosAFF2         PosAFF3         NegAFF1
 Min.   :1.000   Min.   :1.000   Min.   :0.8845
 1st Qu.:2.500   1st Qu.:2.500   1st Qu.:1.0000
 Median :3.000   Median :3.000   Median :1.5000
 Mean   :2.991   Mean   :3.069   Mean   :1.7007
 3rd Qu.:3.500   3rd Qu.:3.500   3rd Qu.:2.0000
 Max.   :4.000   Max.   :4.000   Max.   :4.0000
    NegAFF2         NegAFF3           Sex
 Min.   :0.864   Min.   :0.9186   Min.   :1.000
 1st Qu.:1.000   1st Qu.:1.0000   1st Qu.:1.000
 Median :1.495   Median :1.5000   Median :1.000
 Mean   :1.527   Mean   :1.5448   Mean   :1.487
 3rd Qu.:2.000   3rd Qu.:2.0000   3rd Qu.:2.000
 Max.   :4.000   Max.   :4.0000   Max.   :2.000
    gender           ethnicity             race
 Length:380         Length:380         Length:380
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character



   genderf       ethnicityf
 male  :195   Asian    : 38
 female:185   Black    : 19
```

Data Descriptions ...

```
Hispanic:  67
White   :256
```

Data Descriptions

```
library(rockchalk)
summarize(affect)
```

```
$numerics
           Agency1  Agency2  Agency3  Extrin1  Extrin2  Extrin3
0%          1.0000   1.0000   1.0000   0.9717   1.0000   0.9548
25%         2.0517   2.1667   2.1667   1.3190   1.1845   1.1667
50%         2.4938   2.5000   2.5000   1.5000   1.5380   1.5000
75%         2.8333   2.8984   2.8333   1.8333   2.0000   1.9320
100%        4.0000   4.0000   4.0000   3.5215   3.5000   3.8397
mean        2.4418   2.5497   2.5439   1.6151   1.6861   1.6333
sd          0.5161   0.5274   0.5461   0.4806   0.5767   0.5706
var         0.2664   0.2782   0.2982   0.2309   0.3326   0.3256
skewness    0.1408  -0.0358   0.0684   0.9877   0.9253   1.2113
kurtosis    0.0174   0.1741   0.2069   0.7525   0.3598   1.5591
NA's        0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
N         380.0000 380.0000 380.0000 380.0000 380.0000 380.0000
           Intrin1  Intrin2  Intrin3  NegAFF1  NegAFF2  NegAFF3
0%          1.0000   1.0000   1.0000   0.8845   0.8640   0.9186
25%         2.5000   2.5000   2.5000   1.0000   1.0000   1.0000
50%         3.0000   3.0000   3.0000   1.5000   1.4947   1.5000
75%         3.5000   4.0000   4.0000   2.0000   2.0000   2.0000
100%        4.0000   4.0252   4.0772   4.0000   4.0000   4.0000
```

Data Descriptions ...

```
mean          3.0019    2.9872    3.0797    1.7007    1.5271    1.5448
sd            0.7695    0.8516    0.7705    0.7142    0.6632    0.6528
var           0.5921    0.7253    0.5936    0.5101    0.4398    0.4261
skewness     -0.2988   -0.4639   -0.4596    1.2456    1.5074    1.4087
kurtosis     -0.7696   -0.6594   -0.5367    1.4574    2.0909    1.6862
NA's          0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
N           380.0000  380.0000  380.0000  380.0000  380.0000  380.0000
             PosAFF1   PosAFF2   PosAFF3       Sex
0%            1.0000    1.0000    1.0000    1.0000
25%           2.7435    2.5000    2.5000    1.0000
50%           3.0234    3.0000    3.0000    1.0000
75%           3.5000    3.5000    3.5000    2.0000
100%          4.0000    4.0000    4.0000    2.0000
mean          3.1355    2.9906    3.0694    1.4868
sd            0.6677    0.6851    0.7067    0.5005
var           0.4458    0.4693    0.4995    0.2505
skewness     -0.4096   -0.2338   -0.4469    0.0524
kurtosis     -0.5044   -0.4547   -0.3576   -2.0025
NA's          0.0000    0.0000    0.0000    0.0000
N           380.0000  380.0000  380.0000  380.0000

$factors
          ethnicity                  ethnicityf
 White          :256.0000    White          :256.0000
```

Data Descriptions ...

```
 Hispanic        :  67.0000    Hispanic        :  67.0000
 Asian           :  38.0000    Asian           :  38.0000
 Black           :  19.0000    Black           :  19.0000
NA's             :   0.0000   NA's             :   0.0000
 entropy         :   1.3736    entropy         :   1.3736
 normedEntropy:   0.6868       normedEntropy:   0.6868
N                :380.0000    N                :380.0000
          gender                        genderf
 male            :195.0000     male            :195.0000
 female          :185.0000     female          :185.0000
NA's             :   0.0000   NA's             :   0.0000
 entropy         :   0.9995    entropy         :   0.9995
 normedEntropy:   0.9995       normedEntropy:   0.9995
N                :380.0000    N                :380.0000


           race
 White           :256.0000
 Nonwhite        :124.0000
NA's             :   0.0000
 entropy         :   0.9111
 normedEntropy:   0.9111
N                :380.0000
```

Data Descriptions

```
library(psych)
describe(affect)
```

	vars	n	mean	sd	median	trimmed	mad	min	max
Agency1	1	380	2.44	0.52	2.49	2.43	0.50	1.00	4.00
Agency2	2	380	2.55	0.53	2.50	2.55	0.49	1.00	4.00
Agency3	3	380	2.54	0.55	2.50	2.54	0.49	1.00	4.00
Intrin1	4	380	3.00	0.77	3.00	3.04	0.74	1.00	4.00
Intrin2	5	380	2.99	0.85	3.00	3.05	0.77	1.00	4.03
Intrin3	6	380	3.08	0.77	3.00	3.14	0.74	1.00	4.08
Extrin1	7	380	1.62	0.48	1.50	1.56	0.49	0.97	3.52
Extrin2	8	380	1.69	0.58	1.54	1.62	0.55	1.00	3.50
Extrin3	9	380	1.63	0.57	1.50	1.56	0.49	0.95	3.84
PosAFF1	10	380	3.14	0.67	3.02	3.18	0.71	1.00	4.00
PosAFF2	11	380	2.99	0.69	3.00	3.01	0.74	1.00	4.00
PosAFF3	12	380	3.07	0.71	3.00	3.12	0.74	1.00	4.00
NegAFF1	13	380	1.70	0.71	1.50	1.59	0.74	0.88	4.00
NegAFF2	14	380	1.53	0.66	1.49	1.40	0.73	0.86	4.00
NegAFF3	15	380	1.54	0.65	1.50	1.43	0.74	0.92	4.00
Sex	16	380	1.49	0.50	1.00	1.48	0.00	1.00	2.00
gender*	17	380	NaN	NA	NA	NaN	NA	Inf	-Inf
ethnicity*	18	380	NaN	NA	NA	NaN	NA	Inf	-Inf
race*	19	380	NaN	NA	NA	NaN	NA	Inf	-Inf

Data Descriptions ...

```
genderf*       20  380  1.49  0.50    1.00     1.48  0.00  1.00  2.00
ethnicityf*    21  380  3.42  0.97    4.00     3.65  0.00  1.00  4.00
              range   skew  kurtosis    se
Agency1        3.00   0.14      0.02  0.03
Agency2        3.00  -0.04      0.17  0.03
Agency3        3.00   0.07      0.21  0.03
Intrin1        3.00  -0.30     -0.77  0.04
Intrin2        3.03  -0.46     -0.66  0.04
Intrin3        3.08  -0.46     -0.54  0.04
Extrin1        2.55   0.99      0.75  0.02
Extrin2        2.50   0.93      0.36  0.03
Extrin3        2.88   1.21      1.56  0.03
PosAFF1        3.00  -0.41     -0.50  0.03
PosAFF2        3.00  -0.23     -0.45  0.04
PosAFF3        3.00  -0.45     -0.36  0.04
NegAFF1        3.12   1.25      1.46  0.04
NegAFF2        3.14   1.51      2.09  0.03
NegAFF3        3.08   1.41      1.69  0.03
Sex            1.00   0.05     -2.00  0.03
gender*        -Inf     NA        NA    NA
ethnicity*     -Inf     NA        NA    NA
race*          -Inf     NA        NA    NA
genderf*       1.00   0.05     -2.00  0.03
ethnicityf*    3.00  -1.58      1.17  0.05
```

Data Descriptions ...

Outline

Are the means of PosAFF1 and PosAFF2 different?

```
t.test(x = affect$PosAFF1, y = affect$PosAFF2)
```

```
  Welch Two Sample t-test

data:  affect$PosAFF1 and affect$PosAFF2
t = 2.9527, df = 757.5, p-value = 0.003247
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.0485643 0.2412378
sample estimates:
mean of x mean of y
 3.135516  2.990615
```

Named arguments x and y represent 2 columns from the data frame

Is PosAFF1 different between males and females?

- Formula "interface" generates a different behavior

```
t.test(PosAFF1 ~ genderf, data = affect)
```

```
  Welch Two Sample t-test

data:  PosAFF1 by genderf
t = -0.47395, df = 368.65, p-value = 0.6358
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.1678770  0.1026691
sample estimates:
  mean in group male mean in group female
           3.119643             3.152247
```

- The tilde symbol means "depends on"
- Default output uses the Welch correction for different variances in subgroups. Results won't match other programs, but can change that by setting `var.equal` parameter

Is PosAFF1 different between males and females? ...

```
t.test(PosAFF1 ~ genderf, data = affect,
    var.equal = TRUE)
```

```
  Two Sample t-test

data:  PosAFF1 by genderf
t = -0.47529, df = 378, p-value = 0.6349
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.1674855  0.1022776
sample estimates:
  mean in group male mean in group female
            3.119643             3.152247
```

Outline

The lm function

- Linear regression

```
lm(PosAFF1 ~ genderf, data = affect)
```

```
Call:
lm(formula = PosAFF1 ~ genderf, data = affect)

Coefficients:
  (Intercept)   genderffemale
       3.1196          0.0326
```

- Note big change in default output, comparing `t.test` to `lm`.
  - `t.test` default output is "verbose"
  - `lm` output is minimal, inadequate
- Instead, we save the output into an object named "m1" (our choice)

```
m1 <- lm(PosAFF1 ~ genderf, data = affect)
```

The lm function ...

- and then we "inspect" that object with a barrage of follow-up functions.
- Most commonly, `summary()`

```
summary(m1)
```

```
Call:
lm(formula = PosAFF1 ~ genderf, data = affect)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1522 -0.4087 -0.1134  0.3804  0.8804

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.11964    0.04786  65.178   <2e-16 ***
genderffemale 0.03260    0.06860   0.475    0.635
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6684 on 378 degrees of freedom
Multiple R^2:  0.0005973,  Adjusted R^2:  -0.002047
```

The lm function ...

```
F-statistic: 0.2259 on 1 and 378 DF,  p-value: 0.6349
```

- `summary()` is a generic function, there are different “implementations” customized to the different types of inputs
- What other follow-up functions might be used?

anova	analysis of variance style F or likelihood-ratio tests (compare models)
fitted	display predicted values
resid	display residuals

The lm function ...

- The object `m1` has the class "lm" assigned to it, and there are other specialized functions for objects from that class:

```
methods(class = class(m1))
```

```
 [1] add1            alias           anova           case.names
 [5] confint         cooks.distance  deviance        dfbeta
 [9] dfbetas         drop1           dummy.coef      effects
[13] extractAIC      family          formula         hatvalues
[17] influence       kappa           labels          logLik
[21] model.frame     model.matrix    nobs            plot
[25] plotSlopes      predict         print           proj
[29] qqnorm          qr              residuals       rstandard
[33] rstudent        simulate        standardize     summary
[37] variable.names  vcov
see '?methods' for accessing help and source code
```

Syntax: class used in 2 ways? The `class(m1)` function runs "on" the output object m1, and result goes into the `methods()` function's `class` argument.

Add more predictors

```
m2 <- lm(PosAFF1 ~ genderf + Agency1, data =
    affect)
```

"+" sign serves obvious role

```
summary(m2)
```

```
Call:
lm(formula = PosAFF1 ~ genderf + Agency1, data = affect)

Residuals:
     Min       1Q   Median       3Q      Max
-2.21823 -0.40443  0.00502  0.51699  1.27670

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    2.45163    0.16694  14.686  < 2e-16 ***
genderffemale  0.04212    0.06720   0.627    0.531
Agency1        0.27167    0.06516   4.169  3.8e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add more predictors ...

```
Residual standard error: 0.6544 on 377 degrees of freedom
Multiple R^2:  0.04464,  Adjusted R^2:  0.03958
F-statistic: 8.809 on 2 and 377 DF,  p-value: 0.0001824
```

I use influence diagnostics fairly often

```
m2.inf <- influence.measures(m2)
```

The output in m2.inf is voluminous, but we can see just the “suspicious” lines like this

```
summary(m2.inf)
```

```
Potentially influential observations of
  lm(formula = PosAFF1 ~ genderf + Agency1, data = affect) :

    dfb.1_  dfb.gndr  dfb.Agn1  dffit    cov.r    cook.d  hat
26   0.32   -0.11     -0.29      0.33_*  1.00     0.04    0.03_*
38   0.06   -0.03     -0.05      0.06    1.02_*   0.00    0.02
95   0.10    0.04     -0.12     -0.13    1.03_*   0.01    0.03_*
124 -0.01   -0.01      0.01      0.01    1.03_*   0.00    0.02
136  0.14   -0.05     -0.12      0.14    1.03_*   0.01    0.03_*
146 -0.09   -0.03      0.11      0.12    1.03_*   0.01    0.03_*
177  0.00    0.13     -0.05     -0.19    0.96_*   0.01    0.01
193  0.22    0.10     -0.27     -0.32_*  0.99     0.03    0.02
194  0.22    0.10     -0.27     -0.32_*  0.99     0.03    0.02
219  0.06   -0.12     -0.06     -0.17    0.98_*   0.01    0.01
245  0.08    0.03     -0.09      0.10    1.03_*   0.00    0.02
```

I use influence diagnostics fairly often ...

```
252 −0.10   0.04    0.10    0.12    1.03_*  0.00   0.03_*
274 −0.11  −0.12    0.12   −0.21    0.97_*  0.01   0.01
275  0.02  −0.14   −0.02   −0.19    0.96_*  0.01   0.01
336  0.08  −0.18   −0.08   −0.27_*  0.92_*  0.02   0.01
365  0.02   0.01   −0.02    0.03    1.03_*  0.00   0.02
376  0.02  −0.14   −0.02   −0.19    0.96_*  0.01   0.01
380 −0.13  −0.12    0.14   −0.22    0.98_*  0.02   0.01
```

I'm usually interested in dfbeta

Formula can include transformations

```
m3log <- lm(PosAFF1 ~ genderf + log(Agency1),
    data = affect)
```

- "Smooths" (AKA "wiggly curves", splines, generalized additive terms) can be added in same way

High Level plot functions in R Base

- functions provided with base R

create a “device”

plot	hist	barplot
plot.default	boxplot	dotchart
matplot	coplot	

- Run “example(hist)”, “example(barplot)”, and so forth
- Run “demo(graphics)”

Low Level plotting functions

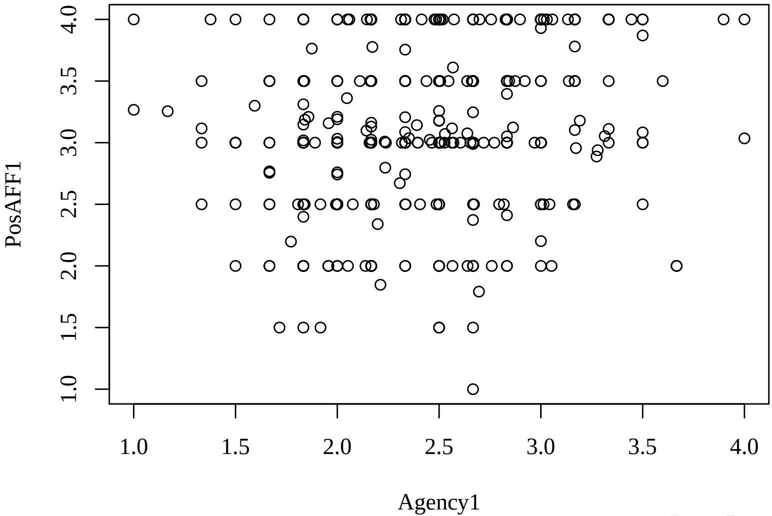
- High level functions create basic plot framework, coordinates
- Low Level functions: added accents or features

text	points	lines	box
arrows	segments	mtext	abline
axis	legend	title	polygon
rect			

Scatterplot

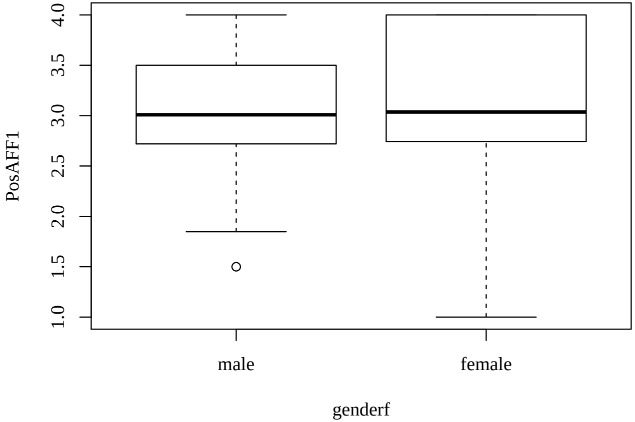
```
plot(PosAFF1 ~ Agency1, data = affect)
```

Scatterplot ...



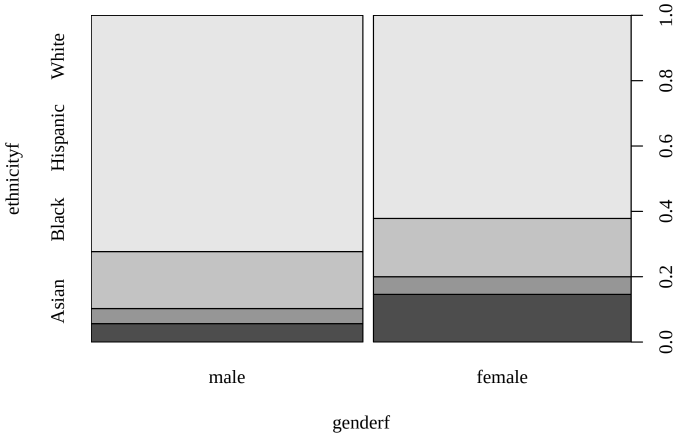
plot is a little bit magic

```
plot(PosAFF1 ~ genderf, data = affect)
```



plot is a little bit magic

```
plot(ethnicityf ~ genderf, data = affect)
```



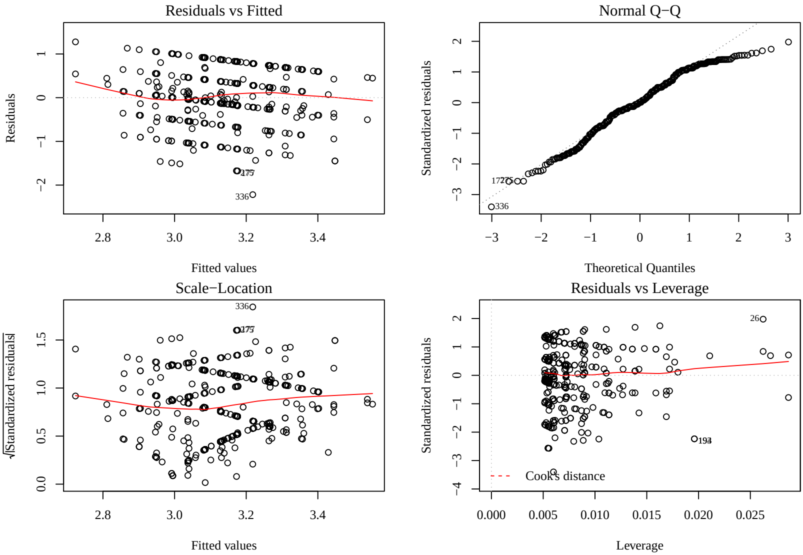
Examples: Regression plot

The `lm` class has a plot method (`plot.lm`)

```
plot(m2)
```

defaults to offer 4 graphs (can be adjusted, see ? `plot.lm`)

Examples: Regression plot ...

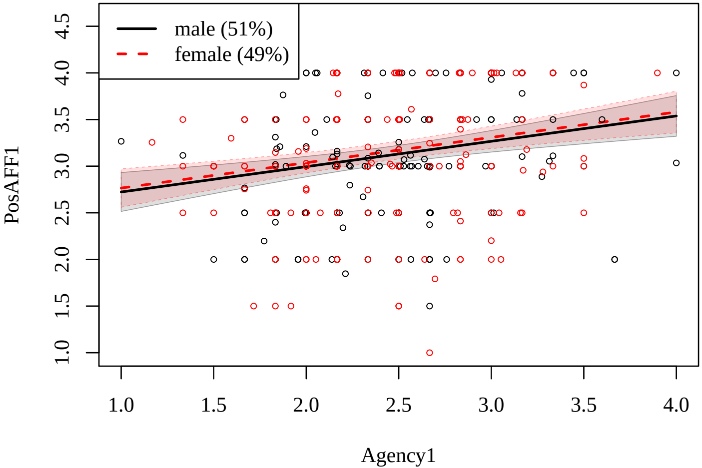


Plot diagnostics

Regression plot from the rockchalk package

```
library(rockchalk)
plotSlopes(m2, plotx = "Agency1", modx =
    "genderf", interval = "confidence")
```

Regression plot from the rockchalk package ...

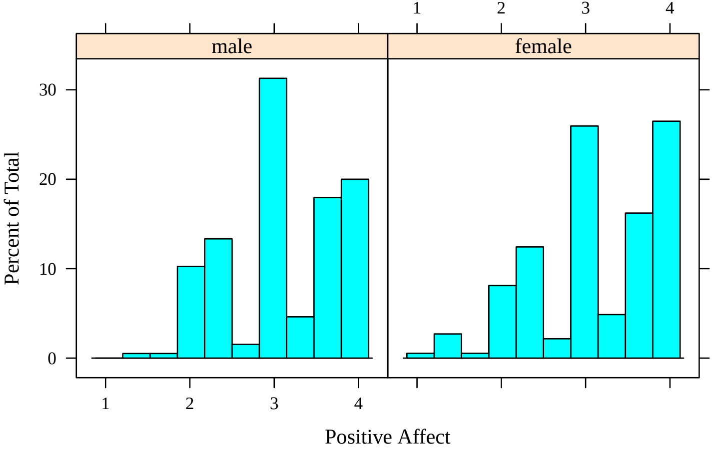


Many plot-oriented packages

- In R's recommended set, the package lattice is intended to produce polished "trellis" plots
- The separate sections are referred to as "panels", which allow intricate customizations
- The formula uses the pipe "|" to signify subgroups

```
library(lattice)
histogram( ~ PosAFF1 | genderf, data = affect,
    xlab = "Positive Affect")
```

Many plot-oriented packages ...



Many plot-oriented packages ...

- A popular package ggplot2 offers similar output under the guise of "facets".

4 Basic Goals Achieved

1. Import data
2. Revise data
3. Do analysis (fit models)
4. Create plots

What's left

1. Project management tools
2. Reproducable document tools
   1. Tools to avoid "pointing and clicking" in report preparation
   2. From R, save result tables and graphs in files

If you ever need help

- Ask in the r-help email list, or in https://stackoverflow.com/questions/tagged/r
- Save some time: Ask your question clearly with
  - Example R code showing what you tried,
  - A copy of the output you think is wrong
  - Output from sessionInfo(), as seen next slide.

Session

```
sessionInfo()
```

```
R version 3.3.3 (2017-03-06)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 17.04

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=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  base

other attached packages:
[1] lattice_0.20-35    psych_1.6.12       rockchalk_1.8.101

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.9         MASS_7.3-45         grid_3.3.3
 [4] MatrixModels_0.4-1  nlme_3.1-131        SparseM_1.74
```

Session ...

```
 [7] minqa_1.2.4        nloptr_1.0.4       car_2.1-4
[10] Matrix_1.2-8       splines_3.3.3      lme4_1.1-12
[13] tools_3.3.3        foreign_0.8-67     pbkrtest_0.4-6
[16] parallel_3.3.3     mnormt_1.5-5       mgcv_1.8-16
[19] nnet_7.3-12        methods_3.3.3      quantreg_5.29
```