

Contessa Wi-Fi bat934393

www.danielezrajohnson.com/rkshop.zip
contains (save to local directory):

- 1) johnson_nwav39_workshop.pdf = these slides
- 2) Rbrul.R = text file of Rbrul program
- 3) rbrul_slide_1... = Rbrul walk-through
- 4) r_slide_1... = R walk-through
- 5) 3 data files = Labov and Becker NYC /r/

thanks for preedback
pair up interrupt raise hand email me

Quantitative analysis with Rbrul and R

bat934393

Daniel Ezra
Johnson

GoldVarb can do:

logistic* regression
with categorical predictors

Rbrul can also do:

logistic* regression
with continuous predictors

linear** regression

interactions between predictors

mixed-effects models



R can also do:

all other statistical techniques

simulations

graphics
for exploratory analysis

graphics
for presentation

Advantages of Rbrul

- flexible in terms of type of response and predictors
- handles typical nested data structure (with mixed models)
- accepts data in non-annoying format(s)
- output in factor weights, log-odds... backward compatible
- some support for interactions between predictors
- runs much faster than GoldVarb
- user may accidentally learn to use R
- “A tool such as Rbrul offers a compromise of the old and the new that I believe will be widely used in the near future.” (Baayen, 2009)
- worth learning, if you like it – it does nothing that special!

Concerns with Rbrul

- how to address problem of multiple comparisons
- how best (not) to use stepwise regression (tomorrow AM)
- cannot test assumptions of models we are building
- does not report standard errors of coefficients
- issues with p-values for mixed models
- how to test for and resolve multicollinearity
- error handling is not good (but customer service is good!)
- programming is not good, difficult to adapt/improve
- like GV, black box, can use w/o understanding what doing
- can get “answers” without thinking about problems
- no pain, no gain? better to learn R?

Basic Rbrul - New York City /r/

Start R.
(Some Windows users may need to right-click and Run As Administrator. Mac users should use R64.)

Load Rbrul.
The usual way to load Rbrul is:
`source("http://www.danielezrajohnson.com/Rbrul.R")`

You can also load a local copy, for example:
`source("~/Linguistics/NWAV 39/NWAV 39 Workshop/Rbrul.R")`

Or you can use the GUI to do this, under the File menu. A browser window will open to track Rbrul use. Close it.

Start Rbrul.
rbrul()
The first time, there will be various messages.
If you get to the MAIN MENU, then everything is working.
If you don't, then there was a problem loading packages.

From the MAIN MENU, choose "1" to "load/save data".
1
If prompted to save current data, press "Enter" for "No".
It will ask "What separates the columns...?"
Enter "c" for commas, because we want to open a .csv file.
c

A file-choosing window appears. Locate and open "ds.csv".

R Console

Type 'demo()' for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help. Type 'q()' to quit R.

[R.app GUI 1.35 (5632) x86_64-apple-darwin9.8.0]
[History restored from /Users/dej/.Rhistory]

```
> source("~/Linguistics/NWAV 39/NWAV 39 Workshop/Rbrul.R")
> rbrul()
```

If you are having trouble downloading the Rbrul project, please visit www.danielezrajohnson.com/project.org. Then launch R, so you can use the GUI option (this option).

No data loaded.

MAIN MENU
1-load/save data
9-reset 0-exit
1: 1

No data loaded.

What separates the columns in the data file to open?
(c-commas s-semicolons t-tabs tf-token file)
Press Enter to exit, keeping current data file, if any.
1: c

Choose File

NWAV 39 Workshop

DEVICES
DEJ's MacBook Pro
Macintosh HD
iDisk

PLACES
Desktop
dej
Applications
Documents

SEARCH FOR
Today
Yesterday
Past Week
All Images

2_solution.R
3_solution.R
4_solution.R
becker_oh_anae.csv
becker_r.csv
ds.csv
Hay Workshop
johnson_...kshop.ppt
johnson_...abstract
nov05
qmul_workshop.pdf
Rbrul.R

Preview:
Name ds.csv
Kind comma-separated values
Size 20 KB on disk
Created 6/2/10 7:14 AM
Modified 6/2/10 7:14 AM
Last opened 6/2/10 7:14 AM

Cancel Open

A screenshot of the top menu bar of the RStudio application. The menu items are: Apple logo, R, File, Edit, Format, Workspace, Packages & Data, Misc, Window, and Help. The 'R' menu is currently selected, and its dropdown menu is visible below it.

The MAIN MENU should appear again, after giving the "Current data structure" as follows:

Current data structure:

```
r (integer with 2 values): 1 0
store (factor with 3 values): Saks Macy's Klein's
emphasis (factor with 2 values): normal emphatic
word (factor with 2 values): fouRth flooR
```

This is the Labov department store data, collected in 1962. The response variable is "r" (presence or absence of /r/). There are three potential predictor variables: "store" (Saks, Macy's, or Klein's), "emphasis" (normal or emphatic), "word" ("fourth" or "floor").

Now we will run a logistic regression with all predictors. Enter "5-modeling" and the MODELING MENU will open.

```
5
Enter "1" to "choose variables".
```

```
1
Enter "1" to choose "r" as the response, or dependent
variable.
```

1 Just press "Enter" as "r" is a binary response.

Enter "2" to choose "1" (presence of /r/) as the application value. We will see what favors and disfavors the presence of /r/. If we used "0" as the application value instead, the model coefficients would be reversed.)

2

```

R Console
~/Linguistics/NWAV 39/NWAV 39 Workshop

1: 1

No data loaded.

What separates the columns in the data file to open?
(c-commas s-semicolons t-tabs tf-token file)
Press Enter to exit, keeping current data file, if any.
1: c

Current data file is: /Users/dej/Linguistics/NWAV 39/NWAV 39 Workshop/
ds.csv

Current data structure:
r (integer with 2 values): 1 0
store (factor with 3 values): Saks Macy's Klein's
emphasis (factor with 2 values): normal emphatic
word (factor with 2 values): fourRth flooR

MAIN MENU
1-load/save data 2-adjust data
4-crosstabs 5-modeling 6-plotting
8-restore data 9-reset 0-exit
1: 5

No variables chosen.

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1: 1
Choose response (dependent variable) by number (1-r 2-store 3-emphasis 4-
word)
1: 1
Type of response? (1-continuous Enter-binary)
1:
Choose application value(s) by number? (1-0 2-1)
1: 2

```

R File Edit Format Workspace Packages & Data Misc Window Help

Enter "2", "Enter", "3", "Enter", "4", "Enter" to choose "store", "emphasis", and "word" as the predictors.
2
3
4

"Are any predictors continuous?" The answer is no, they are all categorical predictors, so just press "Enter".
[Enter]
"Any grouping factors (random effects)?" The answer is no, we will treat these predictors as fixed effects. Unlike many, this data set does not really require a random effect for speaker; it is not coded for it anyway. Press "Enter".
[Enter]
"Consider a(nother) pairwise interaction between predictors?" For now, we will not look at interactions. So just press "Enter".
[Enter]

The MODELING MENU should now show, after we see that Current variables are:
response.binary: r (1 vs. 0)
fixed.factor: store emphasis word

The one-level analysis, similar to the first step of a step-down analysis, is recommended. Enter "2".
2

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

No variables chosen.

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1: 1
Choose response (dependent variable) by number (1-r 2-store 3-emphasis 4-word)
1: 1
Type of response? (1-continuous Enter-binary)
1:
Choose application value(s) by number? (1-0 2-1)
1: 2
Choose predictors (independent variables) by number (2-store 3-emphasis 4-word)
1: 2
2: 3
3: 4
Are any predictors continuous? (2-store 3-emphasis 4-word Enter-none)
1:
Any grouping factors (random effects)? (2-store 3-emphasis 4-word Enter-none)
1:
Consider a(nother) pairwise interaction between predictors? Choose two at a time. (2-store 3-emphasis 4-word Enter-done)
1:

Current variables are:
response.binary: r (1 vs. 0)
fixed.factor: store emphasis word

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1: 2

The first line of the output says:

ONE-LEVEL ANALYSIS WITH store (7.57e-20) + word (5.62e-09)
+ emphasis (0.0661)

The numbers in parentheses are the p-values associated with dropping each predictor from a full model with all of them. These are in order from most to least significant. The notation such as 7.57e-20 is shorthand for 7.57×10^{-20} .

We see that both "store" and "word" are unquestionably statistically significant, while "emphasis" is in a marginal range. In theory, we might collect more data to see if the degree of emphasis really affects /r/ or not.

Next there is a section for each predictor. They are in the order entered, not in order of significance or effect size.

The rightmost column gives factor weights as in GoldVarb. (There is one difference: GoldVarb uses "uncentered" factor weights by default, while Rbrul's default is "centered" factor weights. Uncentered weights depend on the relative numbers of tokens within the factor group. They are highly idiosyncratic and are not recommended.)

The leftmost column has the same coefficients in log-odds units. When the log-odds coefficient x is expressed in "sum contrasts", as here, it relates to the factor weight p by:

$$x = \ln(p/(1-p)) \quad p = \frac{\exp(x)}{(1+\exp(x))}$$

The output also gives the number of tokens for each level

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1: 2

ONE-LEVEL ANALYSIS WITH store (7.57e-20) + word (5.62e-09) + emphasis (0.0661)

\$store

factor	logodds	tokens	1/1+0	centered	factor weight
Saks	0.894	177	0.475		0.71
Macy's	0.454	336	0.372		0.612
Klein's	-1.349	216	0.097		0.206

\$emphasis

factor	logodds	tokens	1/1+0	centered	factor weight
emphatic	0.165	271	0.347		0.541
normal	-0.165	458	0.297		0.459

\$word

factor	logodds	tokens	1/1+0	centered	factor weight
flooR	0.501	347	0.412		0.623
fouRth	-0.501	382	0.228		0.377

\$misc

deviance	df	intercept	grand mean	centered	input prob	Nagelkerke	R2
789.624	5	-0.936	0.316		0.282		0.212

Current variables are:
response.binary: r (1 vs. 0)
fixed.factor: store emphasis word

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1:

The output also gives the number of tokens for each level of the predictor, as well as the raw proportion of the application value for that level - here, labeled "1/1+0".

\$store

factor	logodds	tokens	1/1+0	centered	factor	weight
Saks	0.894	177	0.475			0.71
Macy's	0.454	336	0.372			0.612
Klein's	-1.349	216	0.097			0.206

The employees of the highest-end store, Saks Fifth Avenue, favor post-vocalic /r/ the most (f.w. .710, log-odds .894), and Klein's do the least (.206, -1.349), with Macy's in between (.454, 0.612). (Despite some common practice, we cannot say that factor weights above .500 favor and those below disfavor an alternant. These numbers can only be interpreted relative to the others in the group, or perhaps compared to other studies where the same levels were used.)

For the other predictors, we see that the word "floor" - which bears more stress than "fourth" - is associated with more post-vocalic /r/, and that there is a small effect of emphasis favoring /r/ within repetitions of the same word.

MODELING MENU

1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test

1: 2

ONE-LEVEL ANALYSIS WITH store (7.57e-20) + word (5.62e-09) + emphasis (0.0661)

\$store

factor	logodds	tokens	1/1+0	centered	factor	weight
Saks	0.894	177	0.475			0.71
Macy's	0.454	336	0.372			0.612
Klein's	-1.349	216	0.097			0.206

\$emphasis

factor	logodds	tokens	1/1+0	centered	factor	weight
emphatic	0.165	271	0.347			0.541
normal	-0.165	458	0.297			0.459

\$word

factor	logodds	tokens	1/1+0	centered	factor	weight
flooR	0.501	347	0.412			0.623
fouRth	-0.501	382	0.228			0.377

\$misc

deviance	df	intercept	grand mean	centered	input	prob	Nagelkerke	R2
789.624	5	-0.936	0.316			0.282		0.212

Current variables are:

response.binary: r (1 vs. 0)

fixed.factor: store emphasis word

MODELING MENU

1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test

1:

```

$misc
deviance df intercept grand mean centered input prob
789.624 5 -0.936 0.316 0.282
Nagelkerke R2
0.212

```

For a simple logistic model, the "misc" section gives:
deviance (measure of model non-fit, remembering that
observed 0's and 1's never match the model prediction);
degrees of freedom (1 + k-1 for each predictor);
intercept i (mean predicted log-odds across cells);
grand mean (proportion of response = application value);
centered input probability ($\exp(i)/(1+\exp(i))$);
Nagelkerke pseudo-R-squared value.

We will now look at another New York City /r/ data set.
Enter "9" to return to the main menu.
9

From the MAIN MENU, choose "1" to "load/save data".
1
If prompted to save current data, press "Enter" for "No".
It will ask "What separates the columns...?"
Enter "c" for commas, because we want to open a .csv file.
c

A file-choosing window appears. Open "becker_r.csv".

The MAIN MENU should appear again, with a large number of
variables under "Current data structure".

This is post-vocalic /r/ data collected on the lower East

```

R Console
~/Linguistics/NWAV 39/NWAV 39 Workshop

ONE-LEVEL ANALYSIS WITH store (7.57e-20) + word (5.62e-09) + emphasis
(0.0661)
$store
factor logodds tokens 1/1+0 centered factor weight
Saks 0.894 177 0.475 0.71
Macy's 0.454 336 0.372 0.612
Klein's -1.349 216 0.097 0.206

$emphasis
factor logodds tokens 1/1+0 centered factor weight
emphatic 0.165 271 0.347 0.541
normal -0.165 458 0.297 0.459

$word
factor logodds tokens 1/1+0 centered factor weight
flooR 0.501 347 0.412 0.623
fouRth -0.501 382 0.228 0.377

$misc
deviance df intercept grand mean centered input prob Nagelkerke R2
789.624 5 -0.936 0.316 0.282 0.212

```

Current variables are:
response.binary: r (1 vs. 0)
fixed.factor: store emphasis word

MODELING MENU
1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down
6-plotting 8-settings 9-main menu 0-exit
10-chi-square test
1: 9

Current data file is: /Users/dej/Linguistics/NWAV 39/NWAV 39 Workshop/
ds.csv

Current data structure:
r (factor with 2 values): 1 0

This is postvocalic /r/ data collected by Kara Becker on the Lower East Side of New York around 2007.

We will look at two categorical predictors:

"following" segment: consonant, vowel, or pause;

"age": older or younger speaker. (In general, age and other numeric measures should not be "binned" into categories.)

And we will look at the interaction between them.

This is typical nested sociolinguistic data, so we will use a mixed model with random effects for "speaker" and "word". Without a speaker effect, the significance of "age" will certainly be overestimated, among other possible effects.

Before we start, however, we want to "adjust the data" to only include tokens from casual speech style, which is 77% of this 3000-token data set. One way to do this is to exit Rbrul by entering "0", then enter the following in R:

```
> xtabs(~style,datafile)
> datafile <- datafile[datafile$style == "casual",]
> rbrul()
```

Or from the Rbrul main menu, you can enter "2" to "adjust data", "4" to "retain" certain data, "12" to choose "style", and "1" to retain only casual style tokens.

```
2
4
12
1
[Enter]
```

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

MAIN MENU

```
1-load/save data 2-adjust data
4-crosstabs 5-modeling 6-plotting
8-restore data 9-reset 0-exit
1: 2
```

ADJUSTING MENU

```
1-change class 2-rename 3-exclude 4-retain 5-recode
6-relevel 7-center/transform 8-count 9-main menu 0-exit
10-make interaction group
1: 4
Factor group to retain using? (1-word 2-r 3-time 4-context 5-topic 6-
topic.detail 7-position 8-preceding 9-following 10-syllables 11-
sentential.stress 12-style 13-age 14-education 15-sex 16-speaker 17-
word.type 18-nyc 19-nyce 20-class)
1: 12
Factors to retain from
style? (1-casual 2-reading 3-wordlist)
1: 1
2:
```

ADJUSTING MENU

```
1-change class 2-rename 3-exclude 4-retain 5-recode
6-relevel 7-center/transform 8-count 9-main menu 0-exit
10-make interaction group
1: 9
```

Current data file is: /Users/dej/Linguistics/NWAV 39/NWAV 39 Workshop/ becker_r.csv

Current data structure:

```
word (factor with 501 values): afford affordable after air alarms ...
r (factor with 2 values): 1 0
time (numeric with 1681 values): 1401 2519 2595 1328 1455 ...
context (factor with 2137 values): she can afford to able to afford to
live here I can't afford who couldn't afford the tuition they can't afford
equipment ...
topic (factor with 2 values): neighborhood other
```

Hours later when your plot looks right, use "Save as".
For help on a function, enter "?function".
To search for a function, enter "??function".

12

[Enter]

"Consider a(nother) pairwise interaction between predictors?" To include the "age:following" interaction, type "9", "13".

9

13

[Enter]

The MODELING MENU should now show "Current variables are:"

response.binary: r (1 vs. 0)

fixed.factor: following age

fixed.interaction: following:age

random.intercept: speaker word

From the MODELING MENU, enter "2" for "one-level".

2

A very long output results, because the default setting is to display all the levels of the random effects. Here we note that the standard deviation for words is estimated at 0.765 log-odds and that for speakers is estimated at 0.452.

The first line of the output is:

ONE-LEVEL ANALYSIS WITH speaker (random) + word (random) + following:age (2.32e-07) + following (999) + age (999)

There is only one p-value here. Speaker and word are random effects and are not tested by Rbrul (there are other ways to test their significance). Because the "age:following" interaction is significant, Rbrul does not try dropping "age" or "following". "999" is meant to suggest this.

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

context 5-topic 6-topic.detail 7-position 8-preceding 9-following 10-syllables 11-sentential.stress 13-age 14-education 15-sex 16-speaker 17-word.type 18-nyc 19-nyce 20-class)

1: 9

2: 13

3: 16

4: 1

5:

Are any predictors continuous? (9-following 13-age 16-speaker 1-word Enter-none)

1:

Any grouping factors (random effects)? (9-following 13-age 16-speaker 1-word Enter-none)

1: 16

2: 1

3:

Consider a(nother) pairwise interaction between predictors? Choose two at a time. (9-following 13-age 16-speaker 1-word Enter-done)

1: 9

2: 13

Consider a(nother) pairwise interaction between predictors? Choose two at a time. (9-following 13-age 16-speaker 1-word Enter-done)

1:

Current variables are:

response.binary: r (1 vs. 0)

fixed.factor: following age

fixed.interaction: following:age

random.intercept: speaker word

MODELING MENU

1-choose variables 2-one-level 3-step-up 4-step-down 5-step-up/step-down 6-plotting 8-settings 9-main menu 0-exit 10-chi-square test

1: 2

ONE-LEVEL ANALYSIS WITH speaker (random) + word (random) + following:age (2.32e-07) + following (999) + age (999)

```
following:age (2.32e-07) + following (999) + age (999)
```

There is only one p-value here. Speaker and word are random effects and are not tested by Rbrul (there are other ways to test their significance). Because the "age:following" interaction is significant, Rbrul does not try dropping "age" or "following". "999" is meant to suggest this.

Next are displayed the "main effect" results for following segment and age. Because we are using sum contrasts (treatment contrasts are another option, under "settings"), the results for "following" represent an average over the age categories, and those for "age" represent an average over the following segment categories.

```
$following
```

	factor	logodds	tokens	1/1+0	centered	factor weight
	vowel	1.626	277	0.794		0.836
	consonant	-0.675	1719	0.298		0.337
	pause	-0.952	308	0.260		0.279

A following vowel (.836 or +1.626) favors a post-vocalic /r/ much more than a following consonant (.337 or -0.675). A following pause disfavors /r/ even more (.279 or -0.952).

```
$age
```

	factor	logodds	tokens	1/1+0	centered	factor weight
	younger	0.748	758	0.549		0.679
	older	-0.748	1546	0.257		0.321

Younger speakers favor the use of post-vocalic /r/ (.679 or +0.748) compared to older speakers (.321 or -0.748).

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

```
(2.32e-07) + following (999) + age (999)
```

```
$following
```

	factor	logodds	tokens	1/1+0	centered	factor weight
	vowel	1.626	277	0.794		0.836
	consonant	-0.675	1719	0.298		0.337
	pause	-0.952	308	0.260		0.279

```
$age
```

	factor	logodds	tokens	1/1+0	centered	factor weight
	younger	0.748	758	0.549		0.679
	older	-0.748	1546	0.257		0.321

```
$`following:age`
```

	factor:factor	logodds	tokens	1/1+0	centered	factor weight
	pause:younger	0.695	98	0.602		0.667
	vowel:older	0.678	187	0.770		0.663
	consonant:older	0.017	1149	0.202		0.504
	consonant:younger	-0.017	570	0.493		0.496
	vowel:younger	-0.678	90	0.844		0.337
	pause:older	-0.695	210	0.100		0.333

```
$word
```

	random	logodds	tokens	1/1+0	centered	factor weight
	cars	1.457	7	1.000		0.811
	sure	1.030	15	0.600		0.737
	remember	1.012	35	0.686		0.733
	door	0.887	10	0.900		0.708
	far	0.869	17	0.706		0.704
	barbara	0.838	11	0.364		0.698
	corporation	0.816	2	1.000		0.693
	apartments	0.792	9	0.667		0.688
	picture	0.777	9	0.778		0.685
	percent	0.768	4	0.750		0.683
	marguerite	0.765	5	0.600		0.682
	forward	0.763	2	1.000		0.682
	part	0.758	11	0.727		0.681
	figured	0.757	2	1.000		0.68
	whether	0.690	6	0.833		0.665
	normally	0.685	4	0.750		0.664

The results for the interaction show a coefficient for all six combinations of the interacting variables. This can be misleading because there are only three parameters in a 3x2 interaction such as we have here. The redundancy is seen by each coefficient's having a "mirror image". Using treatment instead of sum contrasts, the numbers would look different.

```
$`following:age`
      factor:factor logodds tokens 1/1+0 centered f.w.
      pause:younger  0.695    98 0.602          0.667
      vowel:older    0.678   187 0.770          0.663
      consonant:older 0.017  1149 0.202          0.504
      consonant:younger -0.017 570 0.493          0.496
      vowel:younger   -0.678    90 0.844          0.337
      pause:older    -0.695   210 0.100          0.333
```

We could parse this by saying that while following consonants behave similarly for the two age groups, the /r/-favoring effect of a following vowel and the /r/-disfavoring effect of a following pause are both two or three times larger for the older speakers.

We get this result by adding the interaction (log-odds) coefficients to those for the main effects. For example, the average following-vowel effect is $+1.626$, but it is $1.626 + 0.678 = 2.304$ for the older speakers compared to $1.626 - 0.678 = 0.948$ for the younger speakers.

These slides have introduced most of the features of Rbrul. If a continuous predictor is used, its coefficient(s) are for a one-unit increase in the predictor. With a continuous response, coefficients are in the units of the response.

R Console

```
~/Linguistics/NWAV 39/NWAV 39 Workshop
C:\SEC 07\... following (555) Page (555)
$following
      factor logodds tokens 1/1+0 centered factor weight
      vowel  1.626    277 0.794          0.836
      consonant -0.675  1719 0.298          0.337
      pause  -0.952    308 0.260          0.279

$age
      factor logodds tokens 1/1+0 centered factor weight
      younger 0.748    758 0.549          0.679
      older  -0.748   1546 0.257          0.321

$`following:age`
      factor:factor logodds tokens 1/1+0 centered factor weight
      pause:younger  0.695    98 0.602          0.667
      vowel:older    0.678   187 0.770          0.663
      consonant:older 0.017  1149 0.202          0.504
      consonant:younger -0.017 570 0.493          0.496
      vowel:younger   -0.678    90 0.844          0.337
      pause:older    -0.695   210 0.100          0.333

$Sword
      random logodds tokens 1/1+0 centered factor weight
      cars  1.457     7 1.000          0.811
      sure  1.030    15 0.600          0.737
      remember 1.012   35 0.686          0.733
      door  0.887    10 0.900          0.708
      far  0.869    17 0.706          0.704
      barbara 0.838   11 0.364          0.698
      corporation 0.816  2 1.000          0.693
      apartments 0.792  9 0.667          0.688
      picture  0.777  9 0.778          0.685
      percent  0.768  4 0.750          0.683
      marguerite 0.765  5 0.600          0.682
      forward  0.763  2 1.000          0.682
      part  0.758   11 0.727          0.681
      figured  0.757  2 1.000          0.68
      whether  0.690  6 0.833          0.665
      normally 0.685  4 0.750          0.664
```


What is R?

- a programming language
- a statistical “environment”
- open-source
- not for everyone? many things are easy, but...
- command-line interface but...
- write scripts
- write functions
- write packages! or at least use packages

some R functions/operators

() [] {} + - * / ^ ! & | %in% %% : = <- == # ? ??

abline	fixef	min	sample
abs	for	mosaicplot	seq
anova	function	names	setwd
as.character	getwd	paste	set.seed
as.factor	glm	pchisq	shapiro.test
as.numeric	glmer	pf	signif
c	grep	plogis	sqrt
cat	head	plot	str
cbind	image	print	summary
class	install.packages	qlogis	table
coef	is.na	ranef	tail
cor	ks.test	range	t.test
data.frame	length	rbind	vector
else	library	read.csv	which
exp	log	rep	wilcox.test
head	logLik	repeat	write.csv
if	max	rnorm	xtabs
ifelse	mean	round	xyplot
fisher.test	median	runif	lm

Basic R - New York City /r/.

Start R.

R has a command-line interface, meaning that you type commands, press Enter, and your commands are executed.

Because of the trial-and-error nature of working with R, most users write and edit their code in a separate file.

On Macintosh, we can choose "New Document" (Command-N) to open such a file. Lines or sections of code will automatically be run in R by choosing "Execute" (Command-Enter). This is a fancy way to cut and paste.

Doing things this way is optional for the workshop. If you are using the command line, note that the up arrow is a very useful way to edit and execute previous commands.

Most R commands will either contain functions, which take arguments within parentheses. Functions sometimes contain other functions. We can also define our own functions.

Try reading the department store data with the `setwd()` and `read.csv()` functions. Use your own directory path.

```
> setwd("~/Linguistics/NWAV 39/NWAV 39 Workshop/")
> read.csv("ds.csv")
```

If this worked, you'll notice that the 729 lines of data were printed to the console. Instead, we want to save the

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

R version 2.12.0 (2010-10-15)
 Copyright (C) 2010 The R Foundation for Statistical Computing
 ISBN 3-900051-07-0
 Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
 You are welcome to redistribute it under certain conditions.
 Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
 Type 'contributors()' for more information and
 'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
 'help.start()' for an HTML browser interface to help.
 Type 'q()' to quit R.

[R.app GUI 1.35 (5632) x86_64-apple-darwin9.8.0]

[History restored from /Users/dej/.Rhistory]

```
> setwd("~/Linguistics/NWAV 39/NWAV 39 Workshop/")
> read.csv("ds.csv")
  r   store emphasis  word
1  1    Saks   normal fouRth
2  1    Saks   normal fouRth
3  1    Saks   normal fouRth
4  1    Saks   normal fouRth
5  1    Saks   normal fouRth
6  1    Saks   normal fouRth
7  1    Saks   normal fouRth
8  1    Saks   normal fouRth
9  1    Saks   normal fouRth
10 1    Saks   normal fouRth
11 1    Saks   normal fouRth
12 1    Saks   normal fouRth
```

were printed to the console. Instead, we want to save the data as an object in the workspace. Use the assignment arrow "<-" to do this. Or use "=", but it's not as cool.

```
> ds <- read.csv("ds.csv")
```

R can read many data types but `.csv` files - text files where the fields are separated by commas, are often used. Microsoft Excel can easily read and write `.csv` files. More general functions are `read.table()` and `readLines()`.

The file is now loaded as a data frame called `"ds"`. A data frame is one type of R object. Other types are vectors, matrixes, and lists. A data frame has named columns, and numbered rows representing observations.

We can see that `ds` is a data frame by:

```
> class(ds)
[1] "data.frame"
```

We could see the entire data frame with `print()`, or by just entering its name. Note that any text after a "#" is a comment, and will be ignored by R. We use comments in our R scripts to make them legible to others, or just to remember what we were trying to do when we wrote them ourselves!

```
> # ds is the same as print(ds), let's not actually run it
```

When we have a large amount of data, we might use `head()` or `tail()` to check the top or bottom of the data.

R Console

~/Linguistics/NWAV 39/NWAV 39 Workshop

Modified	Size
1:42 PM	360 KB
, 2010 3:36 PM	877 KB
, 2010 12:00 PM	9.1 MB
, 2010 1:17 PM	446 KB
, 2010 9:54 AM	53 KB
, 2010 9:53 AM	90 KB
, 2010 9:36 AM	225 KB
, 2010 1:05 PM	2 MB
, 2010 12:20 AM	893 KB
, 2010 11:07 PM	377 KB
, 2010 10:49 PM	664 KB
2, 2010 5:20 PM	18.6 MB

```
>
>
>
>
> ds <- read.csv("ds.csv")
> class(ds)
[1] "data.frame"
> # ds is the same as print(ds), let's not actually run it
>
```

```

> head(ds)

  r store emphasis  word
1 1 Saks    normal fouRth
2 1 Saks    normal fouRth
3 1 Saks    normal fouRth
4 1 Saks    normal fouRth
5 1 Saks    normal fouRth
6 1 Saks    normal fouRth

> tail(ds)

724 0 Klein's emphatic flooR
725 0 Klein's emphatic flooR
726 0 Klein's emphatic flooR
727 0 Klein's emphatic flooR
728 0 Klein's emphatic flooR
729 0 Klein's emphatic flooR

Another function giving a sense of a data file is str(). It
gives information about each column, including its class.

> str(ds)

'data.frame': 729 obs. of  4 variables:
 $ r      : int  1 1 1 1 1 1 1 1 1 ...
 $ store   : Factor w/ 3 levels "Klein's","Macy's",...
 $ emphasis: Factor w/ 2 levels "emphatic","normal"
 $ word    : Factor w/ 2 levels "floR","fouRth"

```

```

R Console
~/Linguistics/NWAV 39/NWAV 39 Workshop

>
>
>
> ds <- read.csv("ds.csv")
>
> class(ds)
[1] "data.frame"
>
> # ds is the same as print(ds), let's not actually run it
>
> head(ds)
  r store emphasis  word
1 1 Saks    normal fouRth
2 1 Saks    normal fouRth
3 1 Saks    normal fouRth
4 1 Saks    normal fouRth
5 1 Saks    normal fouRth
6 1 Saks    normal fouRth
>
> tail(ds)
  r store emphasis  word
724 0 Klein's emphatic flooR
725 0 Klein's emphatic flooR
726 0 Klein's emphatic flooR
727 0 Klein's emphatic flooR
728 0 Klein's emphatic flooR
729 0 Klein's emphatic flooR
>
> str(ds)
'data.frame': 729 obs. of  4 variables:
 $ r      : int  1 1 1 1 1 1 1 1 1 ...
 $ store   : Factor w/ 3 levels "Klein's","Macy's",...: 3 3 3 3 3 3 3 3 3
 $ emphasis: Factor w/ 2 levels "emphatic","normal": 2 2 2 2 2 2 2 2 2
 $ word    : Factor w/ 2 levels "floR","fouRth": 2 2 2 2 2 2 2 2 2 ...
>

```

Make sure that all the columns of your data frame are the correct class. The most common problem arises when you want a column of numbers (class "int[eger]" or "num[erical]") to be treated as a categorical "factor", say by a modeling function. The column named "r" is like this. It actually doesn't matter; column "r" is all 1's and 0's. Still, practice the syntax for changing the class, as follows:

```
> class(ds$r)
[1] "integer"
> ds$R <- as.factor(ds$r)
> class(ds$R)
[1] "factor"
> str(ds)
'data.frame': 729 obs. of  4 variables:
 $ r      : int 1 1 1 1 1 1 1
 $ R      : Factor w/ 2 levels "0","1"...
```

Just now, we created a new column by using the name "ds\$R". We could have overridden the old column by using "ds\$r".

The "\$" operator is useful for referring to named columns. There is a more general syntax for specifying one or more elements of a 2-dimensional object. This is [rows,columns] where rows and columns are specified by number or by name. If either is blank, it means "all rows" or "all columns".

```
> ds[1,] # row 1 (all columns)
> ds[1:3,1] # rows 1-3, column 1 (all rows)
> ds[, "r"] # column named "r" (same as above)
> ds[ds$store=="Macy's",] # all rows where store is Macy's
```

The last example is a very commonly used structure. Note

R Console

```
> str(ds)
'data.frame': 729 obs. of  4 variables:
 $ r      : int 1 1 1 1 1 1 1 1 1 ...
 $ store   : Factor w/ 3 levels "Klein's","Macy's",...: 3 3 3 3 3 3 3 3 3
 3 ...
 $ emphasis: Factor w/ 2 levels "emphatic","normal": 2 2 2 2 2 2 2 2 2
 2 ...
 $ word     : Factor w/ 2 levels "floR","fouRth": 2 2 2 2 2 2 2 2 2 ...
>
> class(ds$r)
[1] "integer"
>
> ds$R <- as.factor(ds$r)
>
> class(ds$R)
[1] "factor"
>
> str(ds)
'data.frame': 729 obs. of  5 variables:
 $ r      : int 1 1 1 1 1 1 1 1 1 ...
 $ store   : Factor w/ 3 levels "Klein's","Macy's",...: 3 3 3 3 3 3 3 3 3
 3 ...
 $ emphasis: Factor w/ 2 levels "emphatic","normal": 2 2 2 2 2 2 2 2 2
 2 ...
 $ word     : Factor w/ 2 levels "floR","fouRth": 2 2 2 2 2 2 2 2 2 ...
 $ R       : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 ...
>
> ds[1,]
  r store emphasis word R
1 1 Saks    normal  fouRth 1
>
> ds[1:3,1]
[1] 1 1 1
>
> # ds[, "r"] (long output)
>
> # ds[ds$store=="Macy's",] (long output)
>
```


The last example is a very commonly used structure. Note the double equals sign. It is also common to forget the comma, causing an error message. However, in the next case, no comma is necessary because we are selecting from a one-dimensional (vector) object. Note use of logical AND ("&").

```
> ds$r[ds$store=="Macy's" & ds$word=="floor"]
```

The function `xtabs()` produces a cross-tabulation. At least on the Macintosh, typing the function name plus the open parentheses causes some information about the function's argument structure to display at the bottom of the window. `xtabs()` requires a formula of the type "a ~ b (+ c ...)" Here we leave the left half blank and put "`ds`" as the data.

```
> xtabs(~ r + store + word, ds)
```

If we want the cross-tab to show the proportion of /r/ in each cell, we take advantage of the fact that column "r" (not column "R"!) is numeric 0's and 1's. We divide one `crosstab` counting the 1's by another counting all tokens. This returns a silly number of decimal places, so we round.

```
> xtabs(r ~ store + word, ds) / xtabs(~ store + word, ds)
> round(xtabs(r~store+word, ds)/xtabs(~store+word, ds), 3)
```

Or, to fit a logistic regression model, we can use `glm()`.

```
> glm(r ~ store + word, ds, family = binomial)
```

`glm()` uses log-odds and treatment contrasts by default.

R Console

```
~/Linguistics/NWAV 39/NWAV 39 Workshop
```

```
, , word = flooR
store
r Klein's Macy's Saks
0 92 82 30
1 12 79 52

, , word = fouRth
store
r Klein's Macy's Saks
0 103 129 63
1 9 46 32

> xtabs(r ~ store + word, ds) / xtabs(~ store + word, ds)
word
store flooR fouRth
Klein's 0.11538462 0.08035714
Macy's 0.49068323 0.26285714
Saks 0.63414634 0.33684211
> round(xtabs(r~store+word, ds)/xtabs(~store+word, ds), 3)
word
store flooR fouRth
Klein's 0.115 0.080
Macy's 0.491 0.263
Saks 0.634 0.337
> glm(r ~ store + word, ds, family = binomial)

Call: glm(formula = r ~ store + word, family = binomial, data = ds)

Coefficients:
(Intercept) storeMacy's storeSaks wordfouRth
-1.814 1.773 2.237 -0.987

Degrees of Freedom: 728 Total (i.e. Null); 725 Residual
Null Deviance: 909
Residual Deviance: 793 AIC: 801
>
```


`glm()` uses log-odds, and by default, treatment contrasts. With treatment contrasts, the effects of most levels of a categorical predictor ("Macy's", "Saks"; "fourth") are compared to a (missing) baseline level ("Klein's"; "floor") set to zero. The intercept is evaluated at the baseline level of all variables.

With sum contrasts, used in ANOVA, VARBRUL, etc., the intercept represents a grand mean (mean over cells), and each coefficient estimates a deviation from the mean. The last coefficient is missing from each predictor, but it can be derived because a predictor's coefficients add to zero.

To use sum instead of treatment contrasts, use the following command (use "contr.treatment" to switch back):

```
> options(contrasts=c("contr.sum", "contr.poly"))
> glm(r ~ store + word, ds, family = binomial)
```

Note: the contrasts used does not affect the model fit. If we want to test whether adding the "emphasis" variable makes a significant improvement to the model, we save both model fits as objects, and use `anova()` to compare them.

```
> model.1 <- glm(r ~ store + word, ds, family = binomial)
> model.2 <- glm(r~store+word+emphasis,ds,family=binomial)
> compare.12 <- anova(model.1, model.2)
> compare.12
```

With models fit by `glm()`, `anova()` reports the difference in deviance, which we compare to a chi-squared distribution to

```
R Console
~/Linguistics/NWAV 39/NWAV 39 Workshop

> glm(r ~ store + word, ds, family = binomial)

Call: glm(formula = r ~ store + word, family = binomial, data = ds)

Coefficients:
(Intercept) storeMacy's storeSaks wordfourth
-1.814      1.773      2.237     -0.987

Degrees of Freedom: 728 Total (i.e. Null); 725 Residual
Null Deviance: 909
Residual Deviance: 793 AIC: 801

> options(contrasts=c("contr.sum", "contr.poly"))
> glm(r ~ store + word, ds, family = binomial)

Call: glm(formula = r ~ store + word, family = binomial, data = ds)

Coefficients:
(Intercept) store1 store2 word1
-0.9704 -1.3366 0.4365 0.4935

Degrees of Freedom: 728 Total (i.e. Null); 725 Residual
Null Deviance: 909
Residual Deviance: 793 AIC: 801

>
> model1 <- glm(r~store+word,ds,family=binomial)
> model2 <- glm(r~store+word+emphasis,ds,family=binomial)
> compare.12 <- anova(model.1,model.2)
> compare.12

Analysis of Deviance Table

Model 1: r ~ store + word
Model 2: r ~ store + word + emphasis
  Resid. Df Resid. Dev Df Deviance
1      725      793.00
2      724      789.62 1      3.378
>
```



With models fit by `glm()`, `anova()` reports the difference in deviance (otherwise known as $-2 \times \log\text{-likelihood}$), which we compare to a chi-squared distribution to obtain a p-value.

```
> pchisq(compare.12$Deviance,compare.12$Df,lower.tail=F)
```

With $p = 0.066$, we might say emphasis is marginally significant. This is the same result as with Rbrul. Indeed, Rbrul arrives at the result by executing the same commands.

Using R, we can create a vast number of types of custom plot. Basic scatterplots can be created with `plot()`, while the packages `lattice` and `ggplot2` are more advanced.

Since the current data is binary, basic scatterplots are not very useful. We could plot 2 "mosaic plots" like this:

```
> par(mfrow=c(1,2)) # sets up for 2 side-to-side plots
> mosaicplot(xtabs(~store+r,ds[ds$word=="fouRth",]))
> mosaicplot(xtabs(~store+r,ds[ds$word=="flooR",]))
```

Functions exist to do many things, but some people may prefer to waste time by hacking a solution. A line chart:

```
> par(mfrow=c(1,1)) # back to one plot per figure
> xt <- xtabs(r ~ store+word,ds) / xtabs(~ store+word, ds)
> plot(1:3,c(0,0,1),col="transparent",xaxt="n",
      xlab="store",ylab="proportion of /r/")
> points(1:3,xt[,1],pch=1,type="b")
> points(1:3,xt[,2],pch=19,type="b")
> axis(1,at=1:3,labels=rownames(xt))
```

[illegible]

```

> par(mfrow=c(1,1)) # back to one plot per figure
> xt <- xtabs(r~store+word,ds)/xtabs(~store+word,ds) # step 1
> plot(c(1,2,3),c(0,0,1),col="transparent",xaxt="n",
      xlab="store",ylab="proportion of /r/") # step 2
> points(1:3,xt[,1],pch=1,type="b") # step 3
> points(1:3,xt[,2],pch=19,type="b") # step 3
> axis(1,at=1:3,labels=rownames(xt)) # step 4
> points(c(1.3,1.3),c(0.8,0.85),pch=c(1,19)) # step 5
> text(c(1.5,1.5),c(0.8,0.85),c("flooR","fouRth")) # step 5

```

In step 1 we get the /r/ proportions by dividing `crosstabs`. In step 2 we use `plot()` to create the plot window with invisible points ensuring the correct dimensions, and label the axes. Note arguments `"col"` for color, `"xaxt"` for x-axis type (`"n"` for none), `"xlab"` and `"ylab"` for axis labels.

In step 3 we use `points()` to plot the real data. Both `plot()` and `points()` require two arguments, the vector of x-coordinates (here 1:3), and the vector of y-coordinates. The argument `"pch=1"` plots unfilled circles and `"pch=19"` plots filled circles. `"type='b'"` means both the points themselves, and connecting lines, will be drawn.

In step 4 we add the x-axis with the store labels. In step 5 we create a legend with the word labels. Notice how both points are created with one command, and then both labels are created with one command.

Hours later when your plot looks right, use "Save as". For help on a function, enter `"?function"`. To search for a function, enter `"??function"`.

The screenshot shows an R console window with the following code being executed:

```

> par(mfrow=c(1,1)) # back to one plot per figure
> xt <- xtabs(r ~ store+word,ds) / xtabs(~ store+word, ds)
> plot(1:3,c(0,0,1),col="transparent",xaxt="n",
+      xlab="store",ylab="proportion of /r/")
> points(1:3,xt[,1],pch=1,type="b")
> points(1:3,xt[,2],pch=19,type="b")
> axis(1,at=1:3,labels=rownames(xt))
> 
> points(c(0.3,0.3),c(0.8,0.85),pch=c(1,19))
> points(c(1.3,1.3),c(0.8,0.85),pch=c(1,19))
> text(c(1.5,1.5),c(0.8,0.85),c("flooR","fouRth"))
> 
> ?points
> 

```

The R Help window is open to the `points` function. The title is "Add Points to a Plot". The description states: "points is a generic function to draw a sequence of points at the specified coordinates. The specified character at the coordinates." The usage is shown as `points(x, ...)` with a note that the default S3 method is `points(x, y = NULL, type = "p", ...)`. The arguments section explains that `x` and `y` are coordinate vectors, `type` is a character indicating the plotting type, and further graphical parameters can be supplied. The details section shows the same code that was executed in the console.

some references

Rbrul homepage: www.danielezrajohnson.com/rbrul.html

Rbrul customer service: rbrul.list@gmail.com (this just emails me)

“Getting off the GoldVarb Standard” article:

http://www.danielezrajohnson.com/johnson_2009.pdf

R-sig-ME listserv: stat.ethz.ch/mailman/listinfo/r-sig-mixed-models

FAQ for the above: <http://glmm.wikidot.com/faq>

R-Lang listserv: <http://pidgin.ucsd.edu/mailman/listinfo/r-lang>

Journal of Memory and Language 59 (issue on Emerging Data Analysis)

new lme4 bible (draft): <http://lme4.R-forge.R-project.org/book/>

older testament:

Pinheiro & Bates 2000. Mixed-Effects Models in S and S-PLUS.

other books:

Harrell, Frank E. 2001. Regression Modeling Strategies.

Baayen, R. Harald. 2008. Analyzing Linguistic Data.

more references

more books:

Dalgaard 2008. Introductory Statistics with R.

<http://stackoverflow.com/questions/192369/>

books-for-learning-the-r-language

official R manual

<http://cran.r-project.org/doc/manuals/R-intro.html>

“no real manual (except Google)”

ask your question and include “r help” as a keyword

R Graph Gallery

<http://addictedtor.free.fr/graphiques/>

Josef Fruehwald’s study group notes

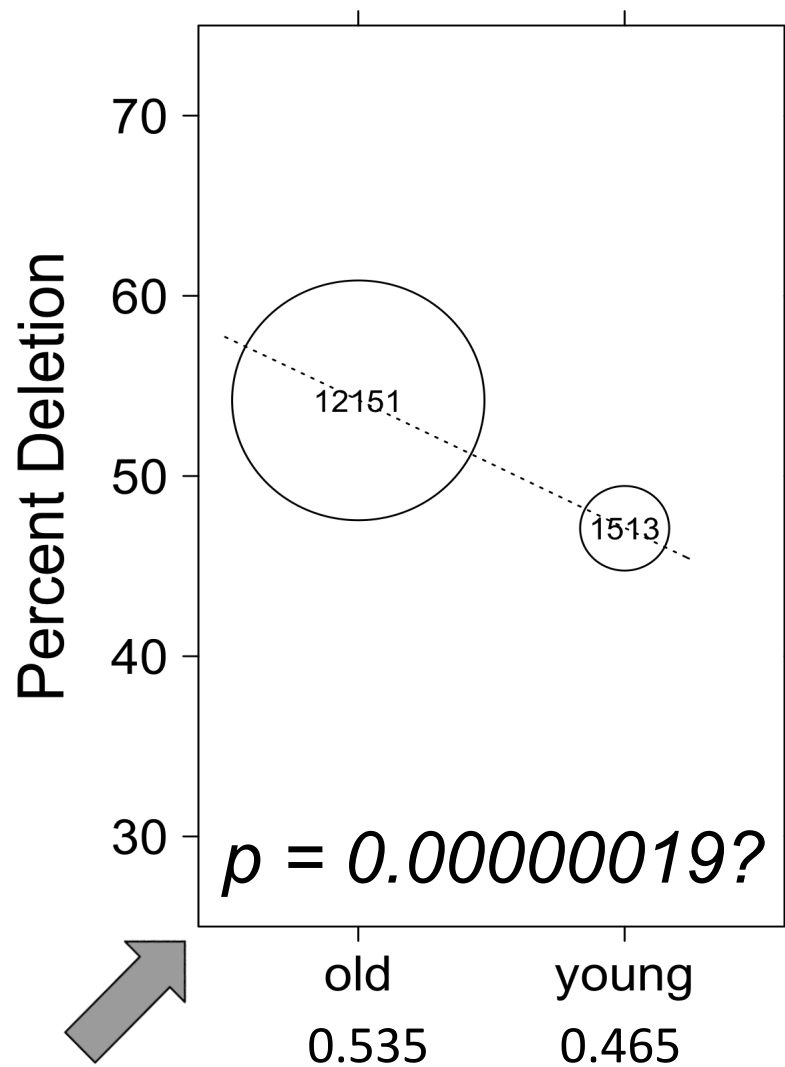
<http://www.ling.upenn.edu/~joseff/rstudy/>

(the next 5 slides are free bonus slides from a previous presentation)

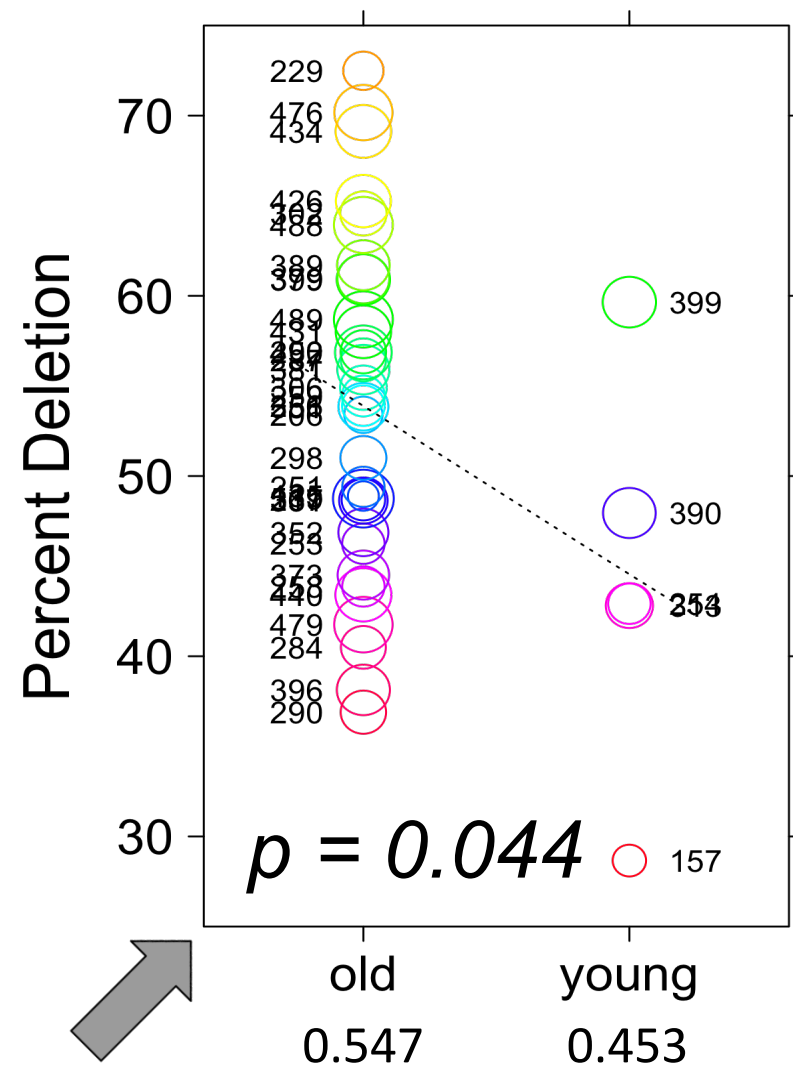
VARBRUL / GoldVarb	other
dependent variable (DV)	DV, response, y
factor group, independent variable (IV)	IV, factor (categorical), predictor, x
factor	level
factor weight	coefficient, effect, estimate, β
factor weight range	similar to 'effect size'
input probability	intercept
applications / total	(response) proportion

lme4	other
mixed model	mixed-effects, hierarchical, or multilevel model
fixed effect	main effect
(all) fixed-effects model	flat model
conditional modes of random effects	random effect estimates, random effect BLUPs

Do you speak VARBRUL?

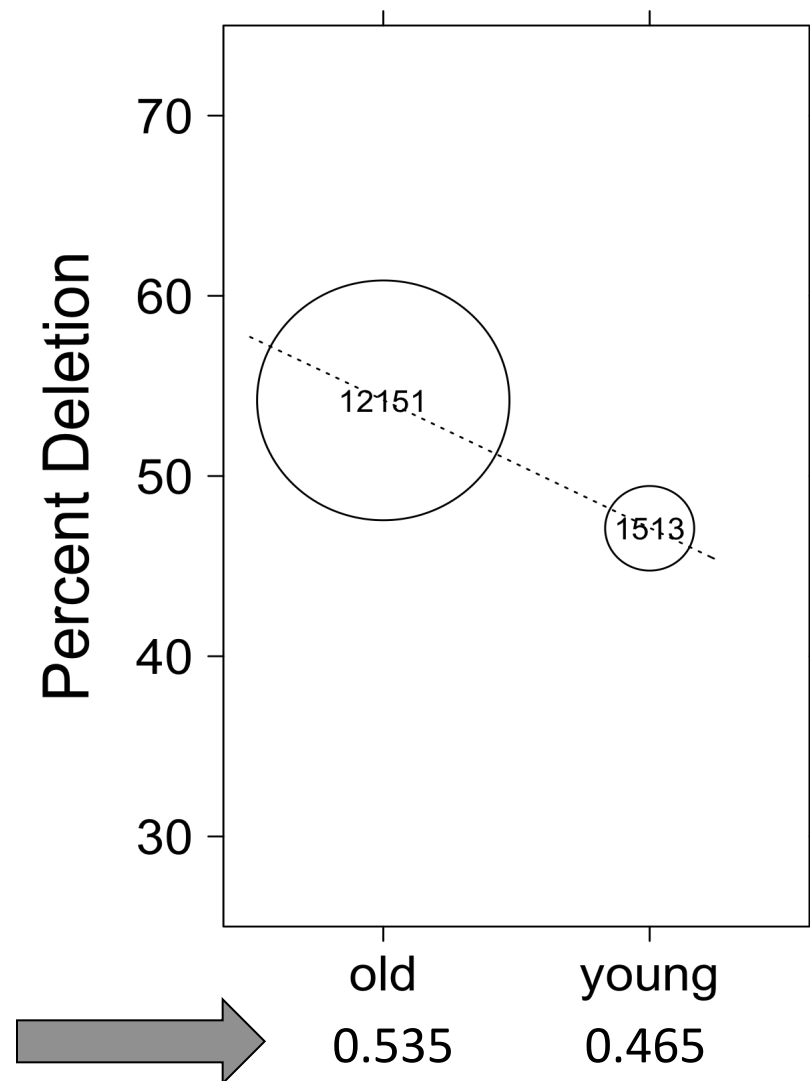


age w/ no random effect

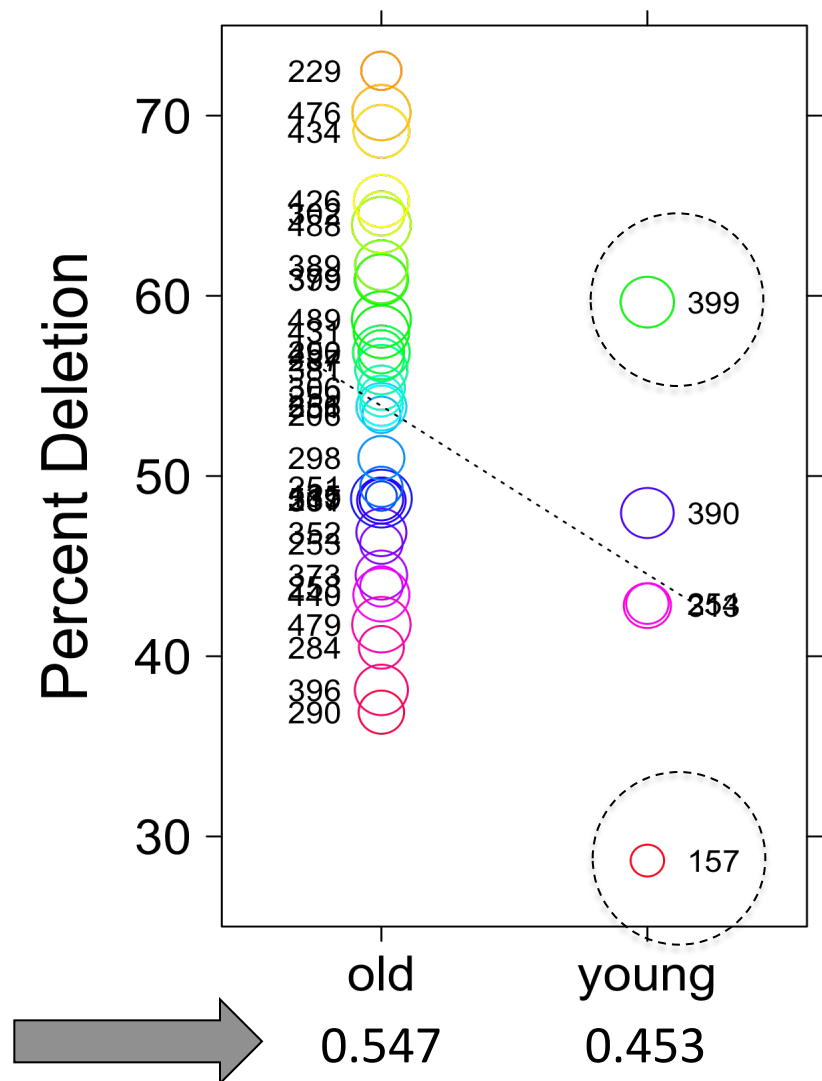


age + random intercept: speaker

Significance of between-speaker predictor

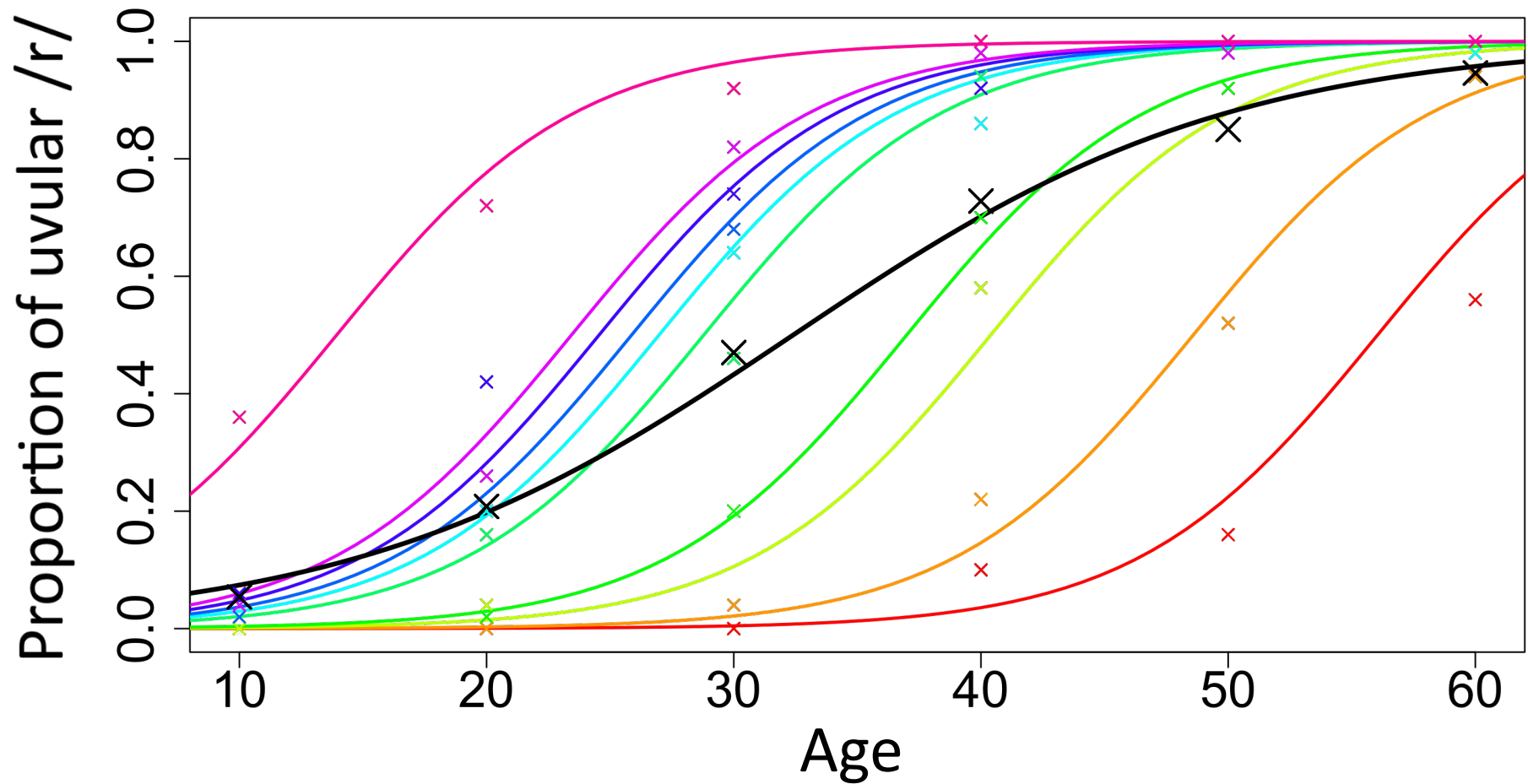


age w/ no random effect



age + random intercept: speaker

Effect size of between-speaker predictor₃₀



age coefficient w/ no random effect: 0.113 log-odds/year
age coeff. w/ speaker random effect: 0.205 log-odds/year

Effect size of within-speaker predictor
(logistic regression only)

between-speaker predictors between-word predictors

constant w/in (data from) each speaker *constant within (data from) each word*
age? gender race class c.o.p. ... frequency gram. cat. int. phon. ..

- significance more accurate:
p = larger, “no longer significant”?
- effect sizes more accurate with
unbalanced data, larger or smaller

within-speaker predictors

vary within (data from) each speaker
age? style phon./gram. context...

word
itself

- effect sizes more accurate:
larger (logistic regression only)

within-word predictors

vary within (data from) each word
stress style ext. phon. ...

speaker
itself

Random effects for speaker & word