

# Using par and cex for graphs

Michael Blasingame

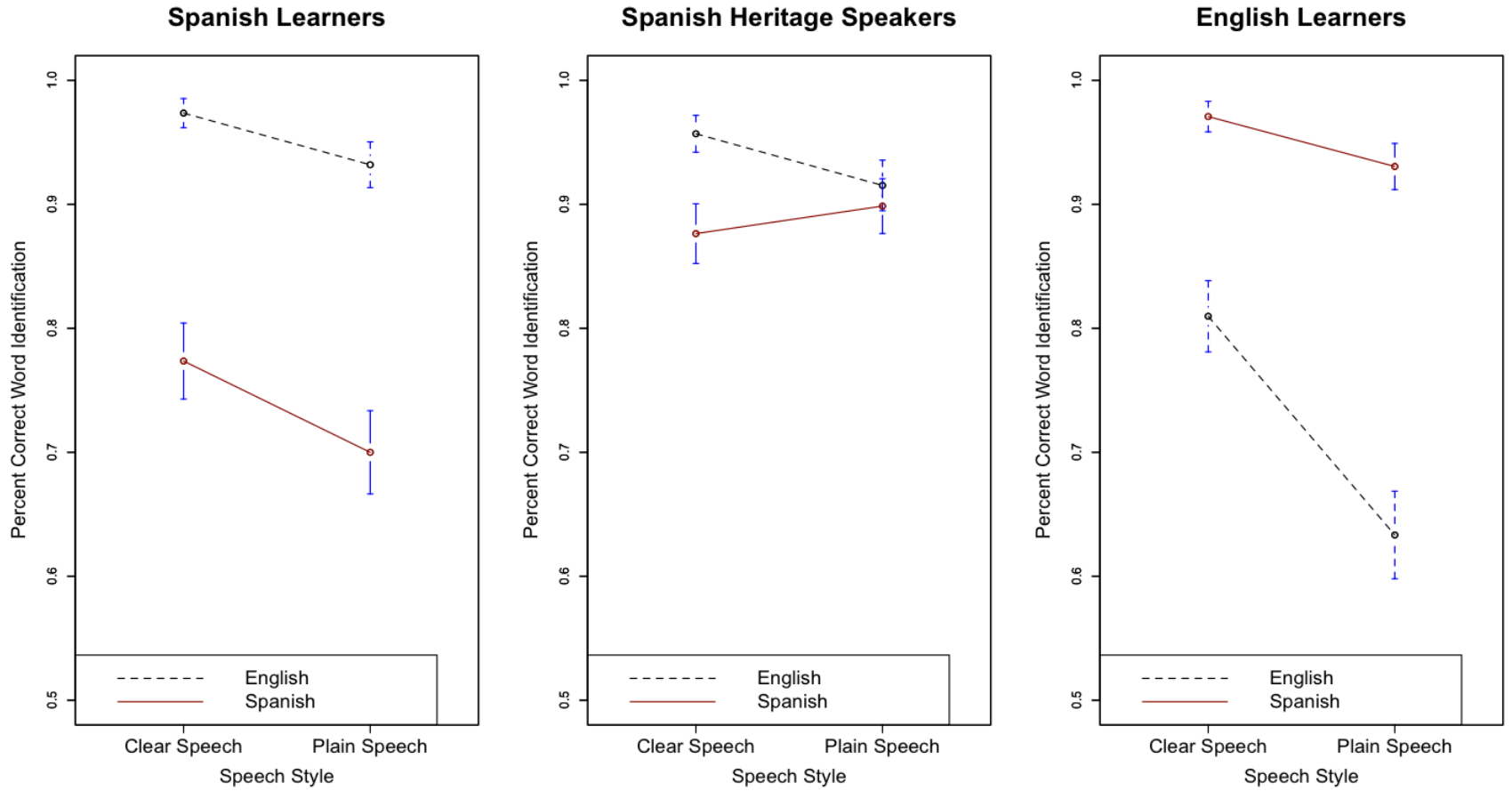
SCRG

2013 May 1

# par

- The function 'par' can be used to manipulate the current plot(s)
- For example, it's most commonly used to allow for multiple plots in one figure
- Code example: `par(mfrow = c(1, 3))`
- Syntax interpretation: Set my figure window to allow a 1 (row) by 3 (column) plotting space

# par(mfrow=c(x,y))



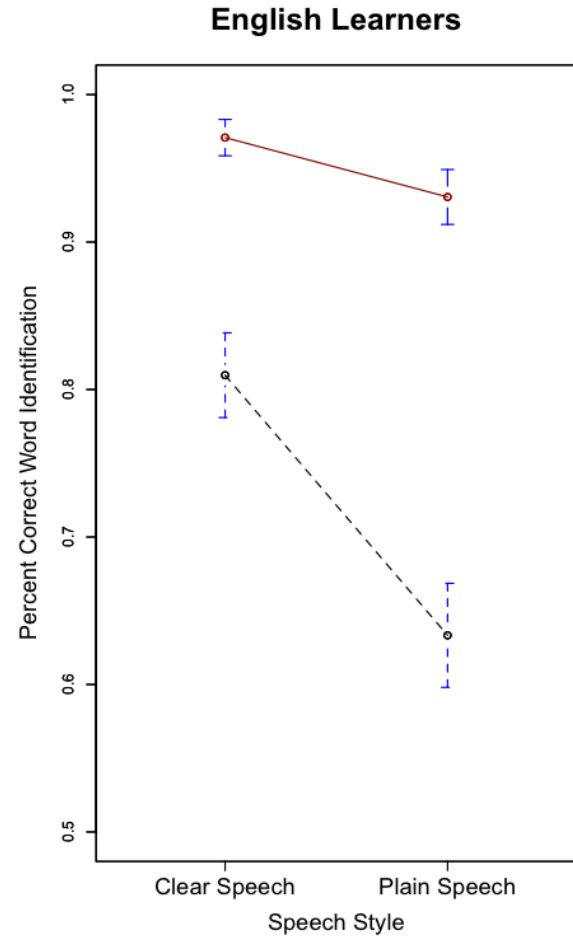
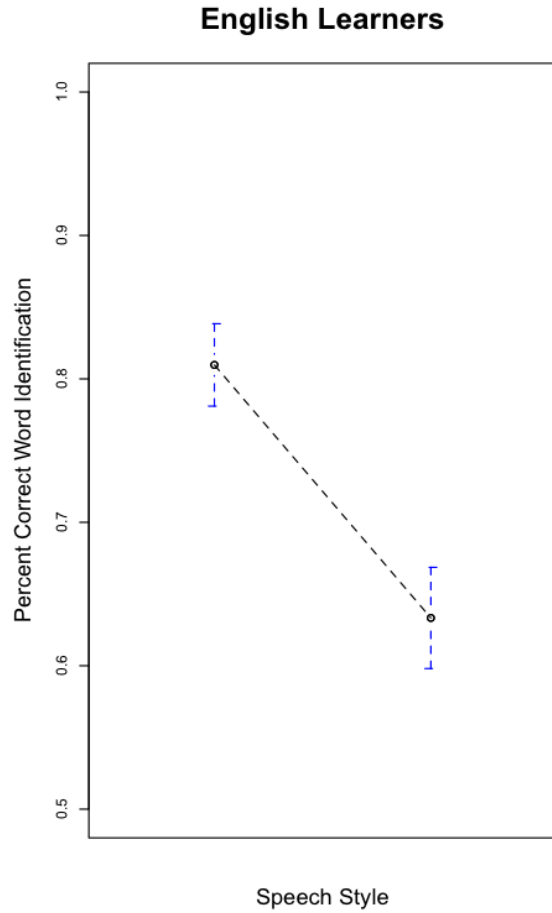
# par(new=T)

- Sometimes, you want to be able to plot two different results on the same plot but can't easily do this in one command due to dataset size
- Example: Plot English and Spanish means of some effect with error bars, different colors, different lines, etc.
- However, R will default to a new plot each time...

# par(new=T)

- The command `par(new=T)` can be useful here
- Syntax interpretation: Keep the plot I just created active—don't create a new one with this next command
- Example (in this order):
  - `plotmeans(data$Correct[data$TaskLang=="ENG" ...])`
  - `par(new=T)`
  - `plotmeans(data$Correct[data$TaskLang=="SPN" ...])`

# par(new=T)



# par(new=T)

- Quick caveat: `par(new=T)` only works if the axes are the same! You can't plot a means plot and then an interaction plot using `par(new=T)`

# `par(mar=c(w,x,y,z))`

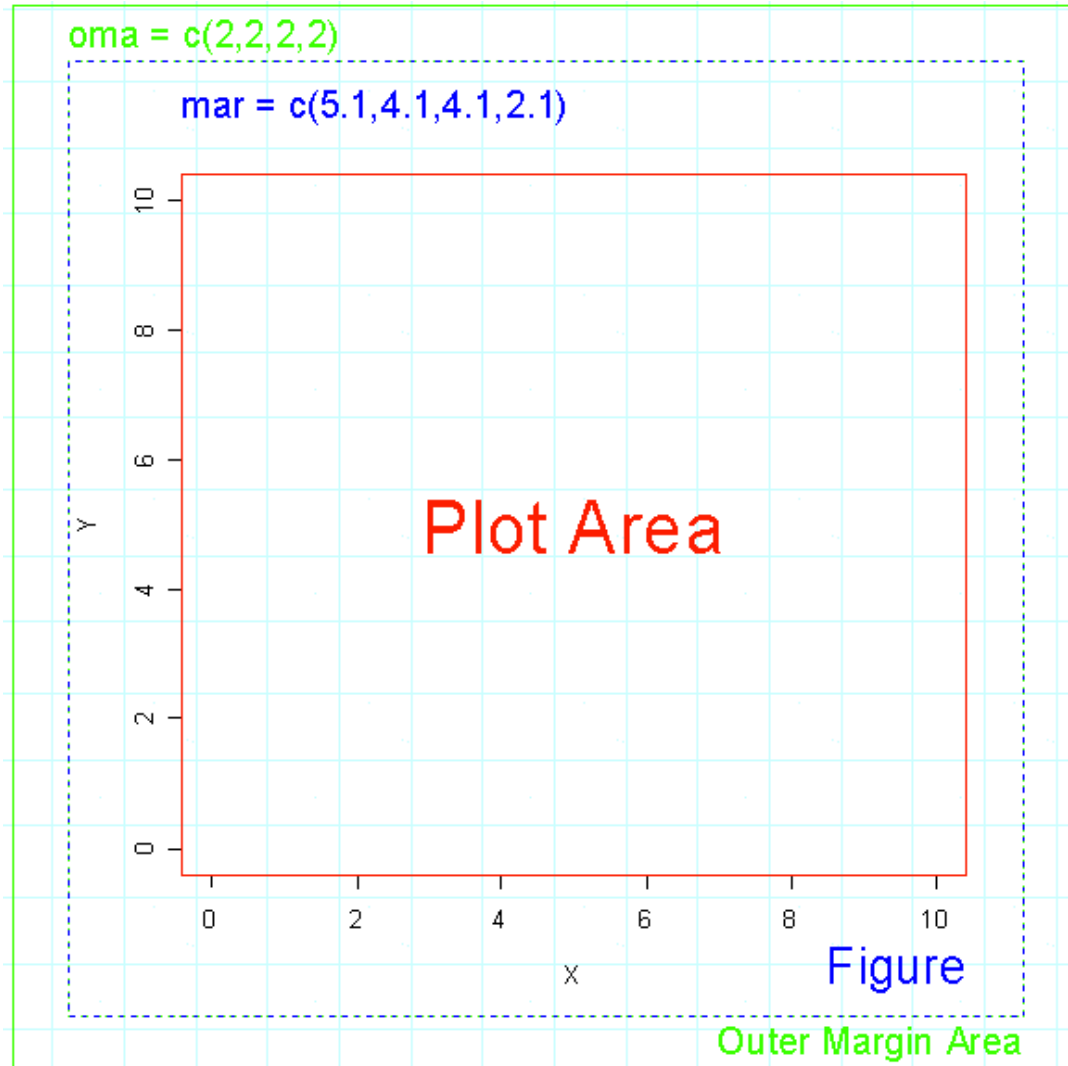
- Sometimes, your axes get crowded
- Using the `par(mar=c(w,x,y,z))` tells the plot how much space the bottom (w), left (x), top (y), and right (z) sides get
- Much of this depends on trial and error (just fiddling a bit until it looks like)



# `par(mar=c(w,x,y,z))`

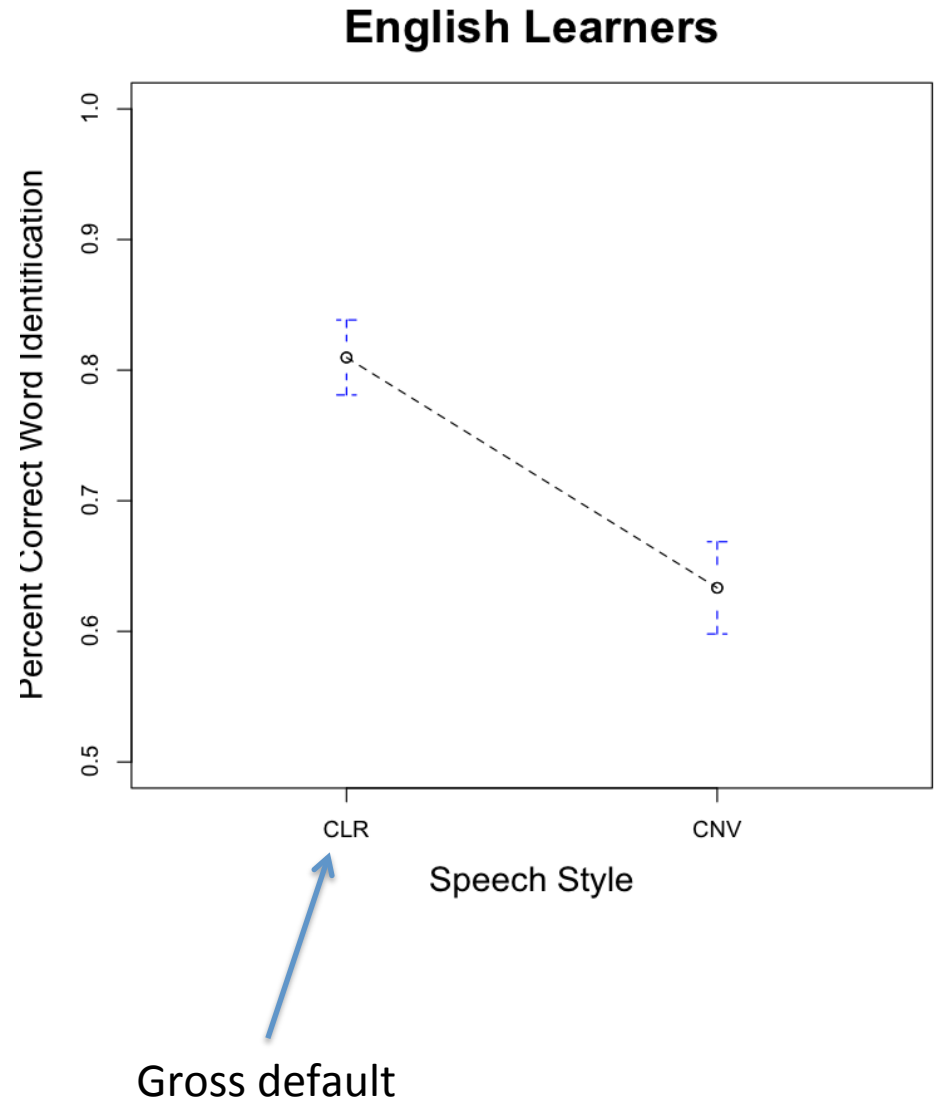
- You can combine `mar` with other `par` features
- Example:
  - `par(mfrow=c(1,3),mar=c(5,5,5,2))`
- Interpretation: Give me a 1 (row) by 3 (column) plot with extra space on the bottom, left, and top (but not a lot of space on the right)

# Area



# axis

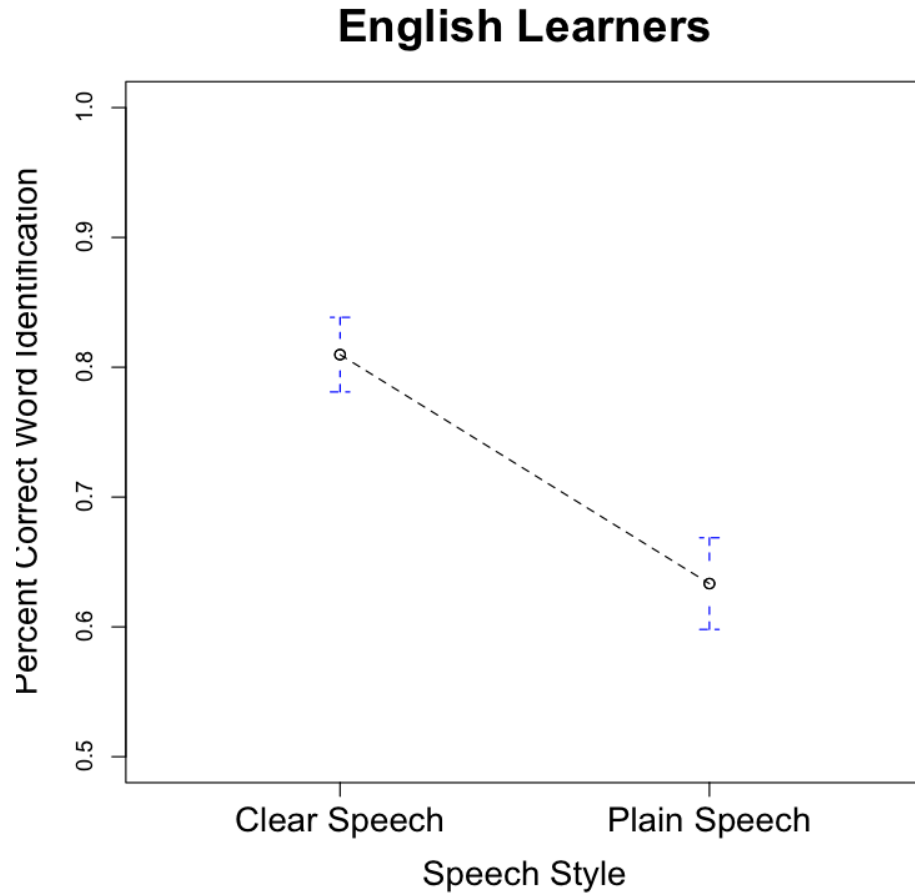
- Sometimes, we want to override the axes labels, because R isn't sure what our unique labeling system means



# axis

- The axis parameter can be useful in overriding where labels are put
- Example:
  - `axis(1.5,1,"Clear Speech",cex.axis=1.5)`
  - `axis(1.5,2,"Plain Speech",cex.axis=1.5)`
- Interpretation: First number indicates where in the vertical space; second number indicates which tick mark. Third is “label”; fourth is size (more on cex in a minute)

# axis



# axis

- Quick caveat: you must turn off your x-axis labels using `xaxt='n'`
- Example:
  - `plot(data...., xaxt='n', main="Title" ...)`

# cex

- The cex family tells you how big to magnify text
- Default is '1'
- Family includes:
  - cex
  - cex.main
  - cex.axis
  - cex.lab

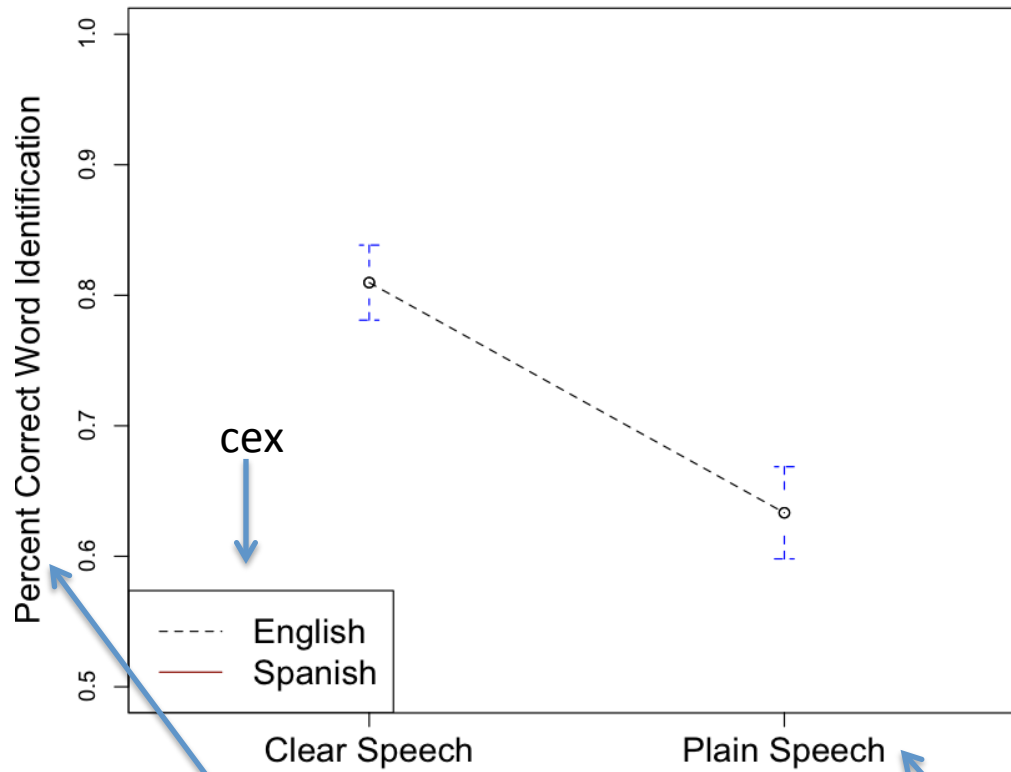
# cex

- cex: for legends and other independent functions outside of the ‘plot’ family
- cex.main: for titles (must have a main=“...”)
- cex.axis: for axes (within the ‘axis’ function)
- cex.lab: for labels within the ‘plot’ family
- Example:
  - plot(... ,cex.lab=1.5,cex.main=2,...)
  - Interpretation: magnify the labels by 1.5 and the title by 2



# cex

English Learners ← cex.main



cex.label

cex.axis

# legend

- Legends orient the reader to your figure without you having to explain each symbol
- Code:
  - ```
legend("bottomleft",lty=c(2,1),col=c("black","dark red"),c("English","Spanish"),cex=1.5)
```
  - Interpretation: legend(“placement”, symbol used, color of symbols, what the different symbols correspond to, magnification parameter)
  - NB: Placement can be “topright”, “topleft”, “bottomright”, “bottomleft”

# legend

- 'legend' only works if there is a current plot open!
- Not to be confused with `par(new=T)`
- Also, notice how 'cex' is used (not 'cex.label', etc.) because 'legend' is its own separate command line (think of it as a mini-figure independent of the rest of the plot)

# plotmeans

- Need library(gplots)
- Calculates means of levels within a factor and plots mean + 95% confidence interval
- User can specify CI confidence level (defaults to  $p = .95$ ) and a variety of other parameters

# plotmeans

- Example:
  - `plotmeans(data$DependentVariable~data$Factor, lty=2, col="darkred", main="Title", xlab="Factor", ylab="Response", ylim=c(minDV,maxDV) ,ci.label=T, n.label=T, p = .95)`
  - Interpretation: `plotmeans`(dependent variable, factor (each level within the factor is separate), line type, color for line, title, factor group, response, y-axis range, #s for CI, n within each level, confidence level)
  - See slide 3 for example combining 'plotmeans' and 'axis' functions