an introduction to R for epidemiologists

graphics

Charles DiMaggio, PhD, MPH, PA-C

New York University Department of Surgery and Population Health
NYU-Bellevue Division of Trauma and Surgical Critical Care
550 First Avenue, New York, NY 10016

Spring 2015

- http://www.columbia.edu/~cjd11/charles_dimaggio/DIRE/
- Charles.DiMaggio@nyumc.org

Outline

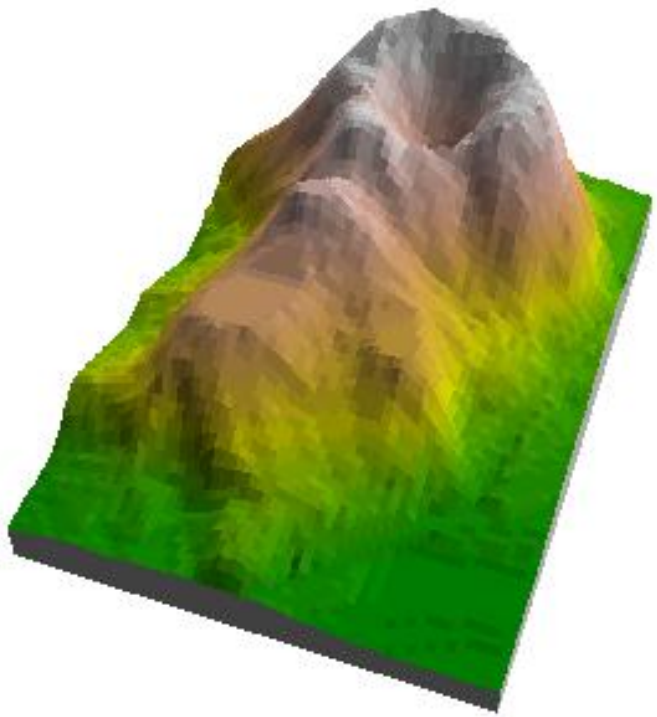
1. Things you can do with R graphics
2. graphing basics
   - about graphing parameters
3. graphing examples
   - side-by-side bar plots
   - a line with confidence limits
   - epidemic curve
   - syphilis
4. ggplot2

Visualizing Facebook Friends

Paul Butler

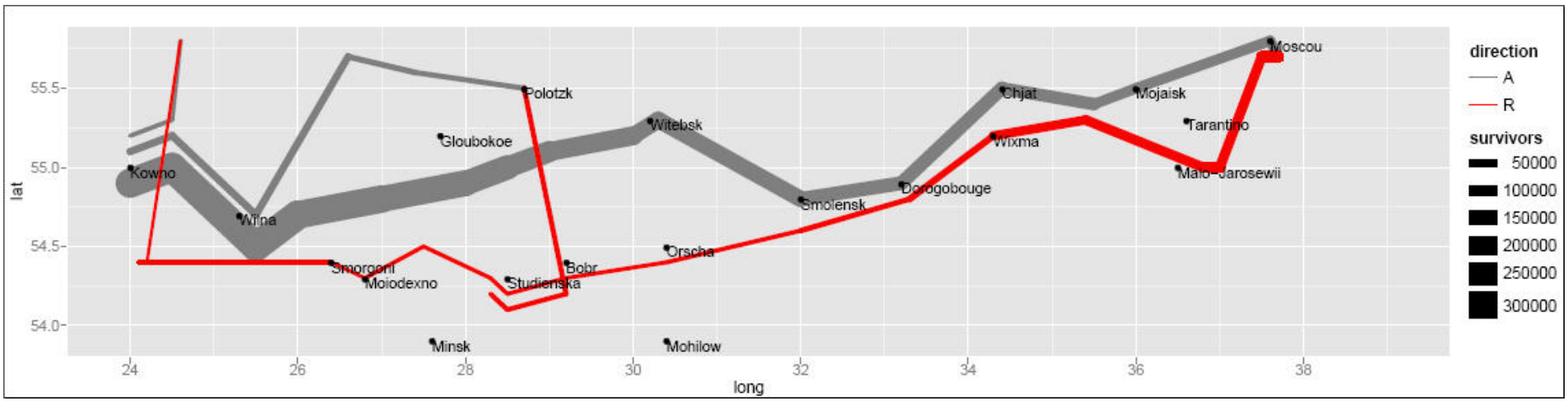
Maunga Whau

R Development Core Team



Napoleon's March to Moscow (Minard)

Hadley Wickham (ggplot2)



Animations

Click on the links

- The Age of Sail
- Wind Map
- Mandelbrot Set

Learn by Example

```
demo(graphics)
example(plot)
example(persp)
```

for many user-written examples

graphing the Duncan data set

John Fox car (companion to applied regression) package

Load the data

```
install.packages("car")
library(car)
data(Duncan)
?Duncan
```

univariate

```
plot(Duncan$income)
plot(Duncan$income, type="l")
plot(Duncan$income, type="h")
plot(Duncan$income, type="h", col="red", lty=3, lwd=5)
```

bivariate

```
plot(Duncan$income, Duncan$education)
abline(lm(Duncan$income~ Duncan$education))
```

add titles and labels

```
plot(Duncan$income, Duncan$education, main="Relationship of
Income and Education", ylab="Education Index",
xlab="Yearly Income (Thousands)")
abline(lm(Duncan$income~Duncan$education), lty=2)
```

R graphic flavors

- base - high-level functions create plots, titles, etc, low-level functions add to existing plots
  - become comfortable with basic plotting before moving on to things like mapping or ggplot2
  - Venables Chapter 12 is your friend
  - Quick-R has a very nice overview
- lattice - (Deepayan Sarkar) based on Trellis from S, allows multi-panels, work on grid
- ggplot2 - (Hadley Wickham) based on "The Grammer of Graphics" (Leland Wilkinson) layer elements to build a graphic
- sp - (Roger Bivand) great package for mapping and spatial analysis (depends on other equally nice packages)

plot()

- generic, high-level, type of plot depends on *class* of arguments
- plot(x,y) - scatterplot
- plot (x) - time series if x is vector, barplot if x is a factor
- plot(f,y) - boxplots if f is factor and y is vector

high-level graphing functions

- hist(x) histograms, lets R choose the breaks
- hist(x, nclass=n) - you choose the number of breaks, (probability=TRUE) bars represent relative frequencies instead of counts
- qqnorm(x) - plots x against normal equivalent
- image(x,y,z) - 3-variable plots, returns a grid
- contour(x,y,z) - returns contour lines,
- persp(x,y,z) - returns 3D image

arguments to high-level graphing functions

- qqline(x) - adds a normal line tp *qqnorm()*
- add=TRUE - adds a high-level plot to an existing high-level plot
- log=x, log=y - plots on logarithmic axes
- axes=FALSE allows you to use the axes() function
- type= - defines type of plot,
  - "p" (point, default), "l" (line), "b" (both)
  - "n" - empty plot, customize with subsequent low-level functions
- main="...", sub= "..." - main and sub titles
- xlab= "...", ylab="..." - axis labels

low-level graphic functions

- points(x,y) , lines(x,y) - add points or lines
- text(x,y,labels=) add text at position x,y
- abline() slope line
  - (a,b) intercept a, slope b
  - (h=y) adds horizontal line
  - (v=x) vertical line
  - (lm=y $\sim$ x) specify least means (regression) object
- legend() adds legend
- title(main, sub)
- axis() use with axes=FALSE from plot() call
- locator() used interactively to select locations with mouse
- identify() - identify data point, e.g. outliers

par()

set parameters (permanent)

- par( ) no arguments returns current parameters, named parameters sets them

```
par(col=4, lty=2)
```

- *save your default parameters before changing*

```
oldpar<-par( )
par(oldpar)# then re-invoke
```

- pass parameter arguments to plot( ) for *temporary* changes

```
plot(x, col=4)
```

- pch - plotting symbol, 1 to 18, to list:

```
plot(0,0, type="n")
legend(0,1, as.character(0:18), pch=0:18)
```

- can use a character as a plotting symbol, e.g pch="A"

basic graphics parameters

- lty - line type
- col - color
- lwd - line width
- cex - character expansion (scales characters smaller or larger)
- font, font.axis, font.label, font.main, font.sub
- mfrow=() mfcol=() - for multiple figures, e.g. mfrow=c(3,2) sets up a 3 by 2 figure

side-by-side bar plot

medicaid births vs total births

```
medicaid <-c(74569, 77344, 76586, 86080,
102088, 109073, 110018)
names (medicaid)<-c("1999", "2000", "2001",
"2002", "2003", "2004", "2005")
barplot(medicaid)
total <-c(255147,258455,253524,
250806,253001,249000,245402)
med.tot<-cbind(medicaid, total) ; med.tot
med.tot.t<-t(med.tot)# transpose to plot correctly
med.tot.t
barplot(med.tot.t, main="",
ylab="Number of Live Births", xlab="Year",
        legend=rownames(med.tot.t), beside=TRUE)
```

plotting a line with upper and lower confidence intervals

- residential proximity to wtc and anxiety
- number of dxs by increasing number of miles from WTC holding the other variables constant at their median values
- 4 sets of numbers: lower limit, point estimate, upper limit and variable against which to plot (in this case miles)

read in data

```
#miles from WTC
miles<-c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24,
     26, 28, 30)
# point estimate
est<-c(570.7778, 527.9492, 488.3342, 452.1437, 418.2168,
386.8357, 357.8092, 330.9608, 306.4333, 283.4399, 262.1718,
242.4996, 224.3035, 207.6803, 192.0969)
#lower limit
low<-c(450.7893, 445.4121, 437.4664, 420.7337, 382.9866,
337.6467, 294.4178, 255.6988, 221.6279, 192.2891, 166.6674,
144.4596, 125.3362, 108.6357, 94.16043)
#upper limit
upper<-c(711.2329, 619.5541, 542.398, 483.9589, 456.2313,
445.4121, 437.9041, 431.3846, 425.8129, 421.1546, 416.5473,
412.4026, 408.2991, 404.6409, 400.6147)
```

the plot

```
#main plot
# note zero the axes, add axes titles etc
plot(miles, est, xlab="Distance from WTC in Miles",
 ylab="Number of Anxiety-Related Diagnoses",
  ylim=c(0,700), pch=15)

#add confidence limit lines
points(miles, low)
points(miles, upper)
lines(miles, est)
lines(miles, low)
lines(miles, upper)
```

plotting an epidemic curve

epitools

```
library(epitools)
sampdates <- seq(as.Date("2004-07-15"),
      as.Date("2004-09-15"), 1)
x <- sample(sampdates, 100, rep=TRUE)
xs <- sample(c("Male","Female"), 100, rep=TRUE)
epicurve.weeks(x)
epicurve.weeks(x, strata = xs)
rr <- epicurve.weeks(x, strata = xs, segments = TRUE)
```

syphilis among men who have sex with men

from Michael C. Samuel, "Public Health Graphical Display Using R"

data: lues by hiv, by half-year

```
 ca.msm.sy <- cbind(c( 6, 7,65,47,126,188,343,463,465,394),
                    c( 0, 2,42,57, 67,101,161,227,291,206),
                    c(31,28,22,12, 29, 47, 76, 74, 61, 67))
 dimnames(ca.msm.sy) <- list(period=paste(rep(1999:2003, each =
 2),c("a","b"),sep=""),status=c("HIV+","HIV-","HIV?"))
all<- apply(ca.msm.sy,1,sum) #marginal total
```

plot all cases

```
barplot(all,col="blue")
barplot(all,las=2,col="blue")
```

plot cases with known status

```
all.w.data <- apply(ca.msm.sy[,1:2],1,sum)
per.hiv<- 100*ca.msm.sy[,"HIV+"]/all.w.data
nn<- length(all)
plot(1:nn,per.hiv,xaxt="n",xlab="",ylab="% HIV+",las=2)
axis(side=1,at=1:nn,labels=dimnames(ca.msm.sy)$period,las=2)
plot(1:nn,per.hiv,xaxt="n",xlab="",ylab="% HIV+",las=2,
     ylim=c(0,100))
axis(side=1,at=1:nn,labels=dimnames(ca.msm.sy)$period,las=2)
```

A double-axis figure

plot number cases, extra space on right

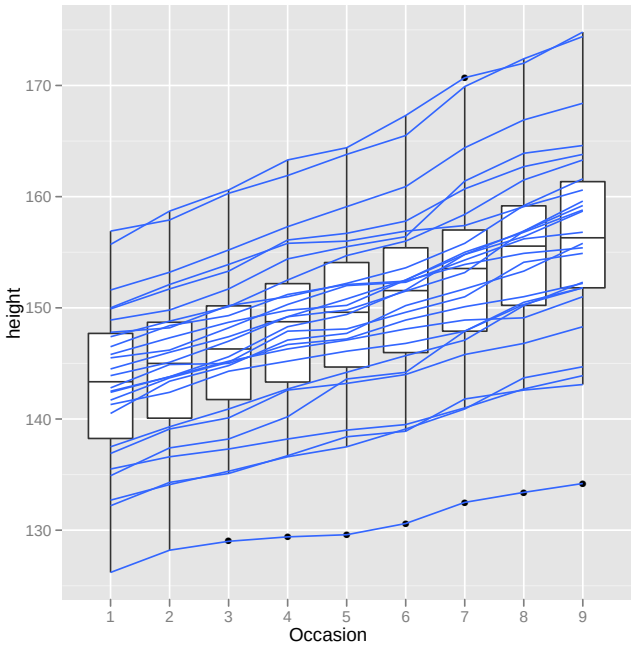
```
parsave <- par()
par(mar=c(5,4,4,4)+.1)
fig1 <- barplot(all,las=2,col="blue")
mtext(side=2,line=3,"Number of Cases",col="blue")
```

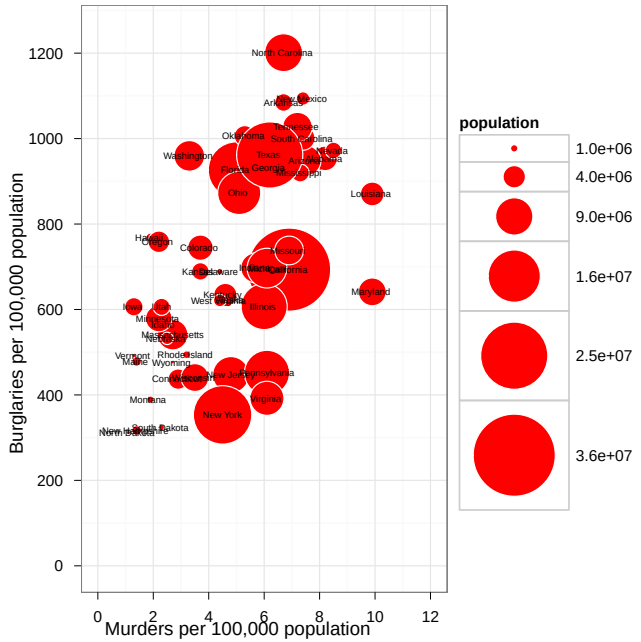
set user parameter for second axis, add lines

```
par(usr=c(par()$usr[1:2],0,100))
lines(fig1,per.hiv,lwd=2,col="red")
axis(side=4,las=2)
mtext(side=4,line=2.5,"Percent HIV-Positive",col="red")
legend(1,40,legend=c("N","%HIV+"),lty=c(NA,2),fil=c(4,NA),col=
red"))
```

the ggplot2 package

- Developed by Hadley Wickham
- Based on the Grammar of Graphics (Wilkinson, 2005)
- Plots built up by adding layers
- Uses (somewhat) idiosyncratic vocabulary
  - data are *mapped* to attributes or *aesthetics* using *geometries* and *scales* and can be displayed as multiple plots or *facets*





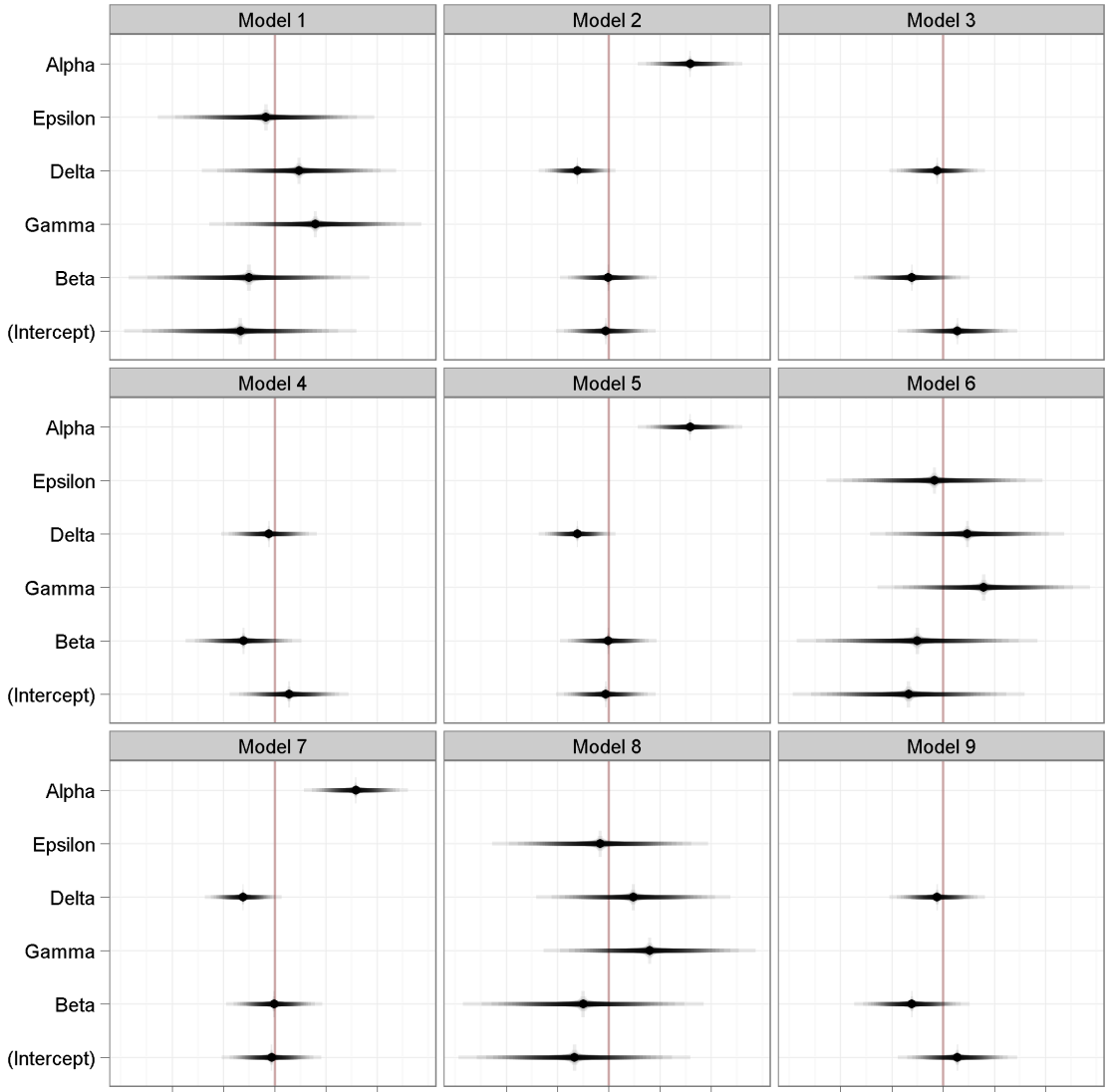
the grammar of graphics

plots built in layers that consist of elements

- geometries - represent data (points, bars, lines)
  - geom_xxx
- aesthetics - colors, shapes, sizes
  - aes()
  - consider whether makes sense for the data (discrete vs. continuous, ordered vs. unordered)
- scales - map geometries to space (linear, logarithmic)
- statistics - identity, mean
- coordinates - how elements represented on page ("canvas")

Regression Coefficients (Faceting)

David Sparks



qplot()

qplot()

1. defaults for quick plotting
2. accepts transformed variables
3. define "aesthetics" by multiple variables categorized by things like color or size

graphically exploring motor trend car tests

```
library(ggplot2)
data(mtcars)
head(mtcars)
?mtcars
qplot(wt, mpg, data=mtcars)
qplot(log(wt), mpg-10, data=mtcars)
qplot(log(wt), mpg-10, data=mtcars, color=qsec)
qplot(log(wt), mpg-10, data=mtcars, color=qsec, size=cyl)
```

save plot and add to it

```
plot1<-qplot(wt, mpg, data=mtcars, geom=c("point", "smooth"))
plot1
plot2<-plot1+facet_wrap(~cyl)
plot2
```

from qplot() to ggplot()

(see Christophe Ladroue)

- more control
- basic form: ggplot() + geom_xxx()
- steps in creating plot
  1. define the data - has to be data frame
  2. add first layer - geometry (plot type) and its aesthetics (variables, categorizing factors)
  3. more layers - scales, facets, titles

```
data(diamonds)
set.seed(53)
small<-diamonds[sample(nrow(diamonds),1000),]

p1<-ggplot(small)
p2<-p1+geom_point(aes(x=carat,y=price,colour=cut))
p2
p2+scale_y_log10()+facet_wrap(~cut)+ggtitle("Cut by Price")
```

geoms, aesthetics, facets

- geoms need **"aesthetics"**
  - shapes defined by name of the geom, e.g. "geom_point"
  - x (and y) variables required
  - color, size, fill, alpha (transparency)
- varied **ggplot syntax**

```
ggplot(small,aes(x=carat,y=price,colour=cut))+geom_point()
ggplot(small,aes(x=carat,y=price))+geom_point(aes(colour=cut))
```

- **other geoms**: geom_smooth for trends

```
p<-ggplot(small,aes(x=carat,y=price))
p+geom_point()+geom_smooth()+facet_wrap(~cut)
p+geom_point()+geom_smooth(method="lm")+facet_wrap(~cut)
```

- **faceting** - to categorize variables
  - facet_wrap() - by single variable

```
p2+facet_wrap(~cut)
p2+facet_wrap(~cut,nrow=1)
p2+facet_wrap(~cut,ncol=1)
```

  - facet_grid - more than one variable

```
p2+facet_grid(cut~color)
```

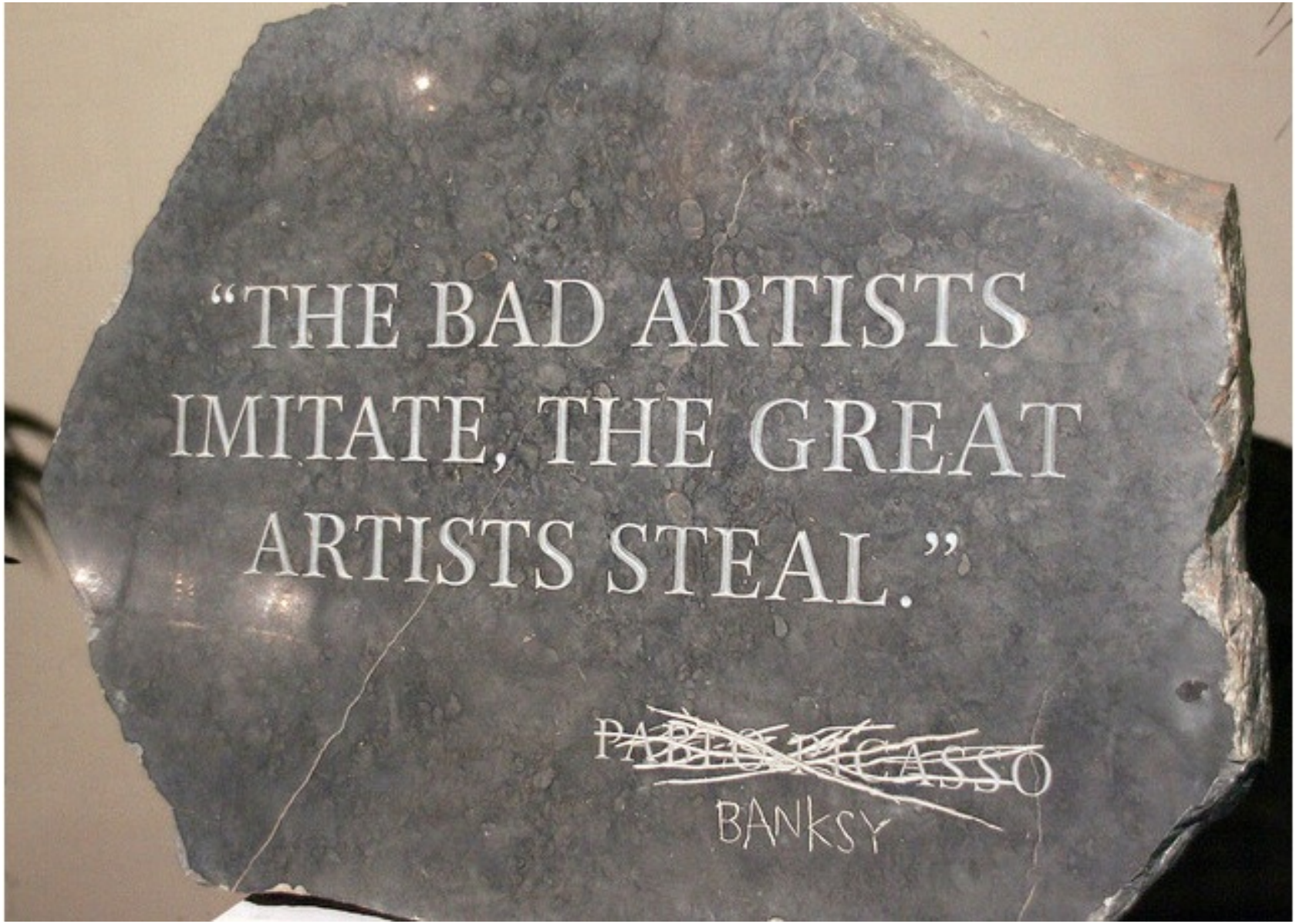


Figure: your turn