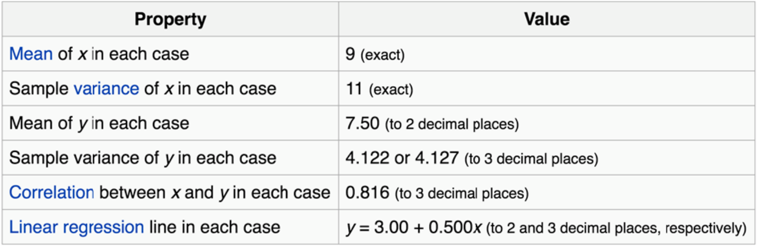
Data visualization in R

Mikhail Dozmorov

Fall 2016

Why visualize data?

- Four groups
- 11 observations (x, y) per group

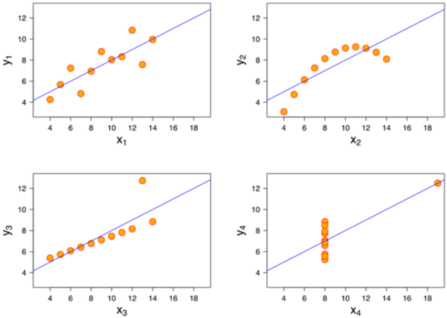


Property	Value
Mean of *x* in each case	9 (exact)
Sample variance of *x* in each case	11 (exact)
Mean of *y* in each case	7.50 (to 2 decimal places)
Sample variance of *y* in each case	4.122 or 4.127 (to 3 decimal places)
Correlation between *x* and *y* in each case	0.816 (to 3 decimal places)
Linear regression line in each case	$y = 3.00 + 0.500x$ (to 2 and 3 decimal places, respectively)

https://en.wikipedia.org/wiki/Anscombe%27s_quartet

Why visualize data?

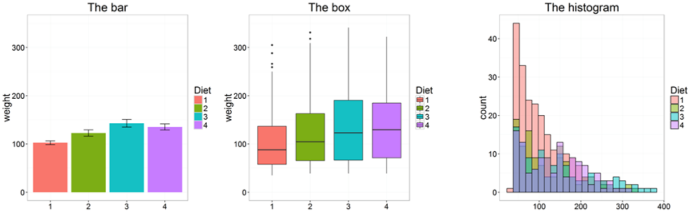
- Four groups
- 11 observations (x, y) per group



https://en.wikipedia.org/wiki/Anscombe%27s_quartet

R base graphics

- `plot()` generic x-y plotting
- `barplot()` bar plots
- `boxplot()` box-and-whisker plot
- `hist()` histograms

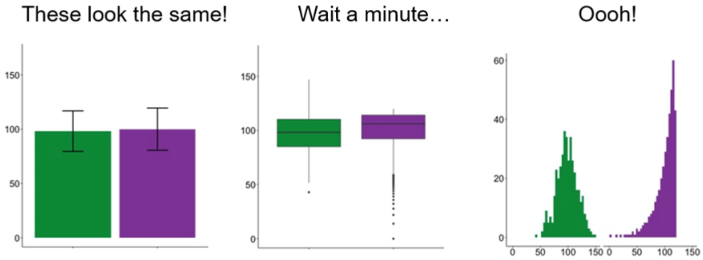

The bar

The box

The histogram

`http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual#TOC-Some-Great-R-Functions`

Don't use barplots



Weissgerber T et.al., "Beyond Bar and Line Graphs: Time for a New Data Presentation Paradigm", PLOS Biology,2015 `http://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1002128`
`https://cogtales.wordpress.com/2016/06/06/congratulations-barbarplots/`

R base graphics

- heatmap() - heatmap

Alternatives:

- qplots::heatmap.2()
- pheatmap::pheatmap()
- NMF::aheatmap()

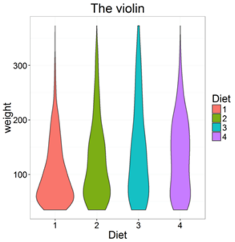
Other useful plots:

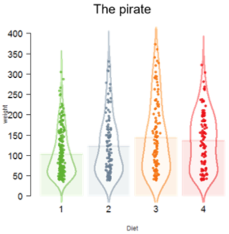
- qqnorm(), qqline(), qqplot() - distribution comparison plots
- pairs() - pair-wise plot of multivariate data

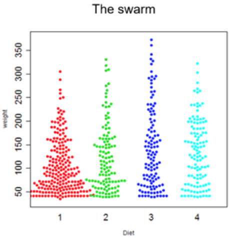
http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual#TOC-Some-Great-R-Functions

Special plots

- `vioplot()`: Violin plot, `https://cran.r-project.org/web/packages/vioplot/`
- `PiratePlot()`: violin plot enhanced. `install_github("ndphillips/yarrr")`, `http://nathanieldphillips.com/`
- `beeswarm()`: The Bee Swarm Plot, an Alternative to Stripchart, `https://cran.r-project.org/web/packages/beeswarm/index.html`







Saving plots

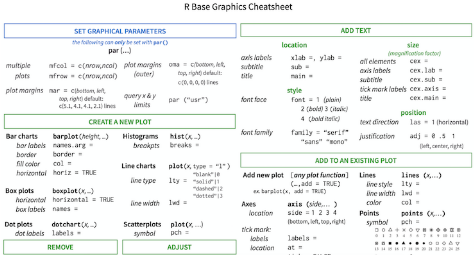
- Save to PDF

```
pdf("filename.pdf", width = 7, height = 5)
plot(1:10, 1:10)
dev.off()
```

- Other formats: `bmp()`, `jpg()`, `pdf()`, `png()`, or `tiff()`
- Learn more `?Devices`

R base graphic cheat-sheet

https://github.com/nbrgraphs/mro/blob/master/BaseGraphicsCheatsheet.pdf



R Base Graphics Cheatsheet

SET GRAPHICAL PARAMETERS

the following can only be set with **par()**

par (...)

multiple plots	mfcol = c(*nrow,ncol*)
	mfrow = c(*nrow,ncol*)
		plot margins (outer)	oma = c(*bottom, left, top, right*) default: c(0, 0, 0, 0) lines

plot margins	mar = c(*bottom, left, top, right*) default: c(5.1, 4.1, 4.1, 2.1) lines	*query x & y limits*	par ("usr")

CREATE A NEW PLOT

Bar charts	**barplot**(*height, ...*)	**Histograms**	**hist**(*x, ...*)
bar labels	names.arg =	*breakpts*	breaks =
border	border =		
fill color	col =	**Line charts**	**plot**(*x*, type = "l")
horizontal	horiz = TRUE	*line type*	lty = "blank"|0 "solid"|1 "dashed"|2 "dotted"|3
Box plots	**boxplot**(*x, ...*)	*line width*	lwd =
horizontal	horizontal = TRUE		
box labels	names =		
Dot plots	**dotchart**(*x, ...*)	**Scatterplots**	**plot**(*x, ...*)
dot labels	labels =	*symbol*	pch =

REMOVE

ADJUST

ADD TEXT

location		**size** *(magnification factor)*	
axis labels	xlab =, ylab =	*all elements*	cex =
subtitle	sub =	*axis labels*	cex.lab =
title	main =	*subtitle*	cex.sub =
style		*tick mark labels*	cex.axis =
font face	font = 1 *(plain)* 2 *(bold)* 3 *(italic)* 4 *(bold italic)*	*title*	cex.main =
		position	
		text direction	las = 1 (horizontal)
font family	family = "serif" "sans" "mono"	*justification*	adj = 0 .5 1 (left, center, right)

ADD TO AN EXISTING PLOT

Add new plot	[*any plot function*] (...,add = TRUE) *ex.* barplot(x, add = TRUE)	**Lines**	**lines** (*x,...*)
		line style	lty =
		line width	lwd =
		color	col =
Axes	**axis** (*side,...*)	**Points**	**points** (*x,...*)
location	side = 1 2 3 4 (bottom, left, top, right)	*symbol*	pch =
tick mark:			
labels	labels =		
location	at =		

Data manipulation

dplyr: data manipulation with R

80% of your work will be data preparation

- getting data (from databases, spreadsheets, flat-files)
- performing exploratory/diagnostic data analysis
- reshaping data
- visualizing data

`http://www.gettinggeneticsdone.com/2014/08/do-your-data-janitor-work-like-boss.html`

dplyr: data manipulation with R

80% of your work will be data preparation

- Filtering rows (to create a subset)
- Selecting columns of data (i.e., selecting variables)
- Adding new variables
- Sorting
- Aggregating
- Joining

http://www.gettinggeneticsdone.com/2014/08/do-your-data-janitor-work-like-boss.html

Dplyr: A grammar of data manipulation

https://github.com/hadley/dplyr

```r
install.packages("dplyr")
```

The pipe %>% operator

- Pipe output of one command into an input of another command - chain commands together
- Think about the "|" operator in Linux
- Read as "then". Take the dataset, *then* do ...

```
library(dplyr)
library(ggplot2)
data(diamonds)
head(diamonds)
diamonds %>% head
summary(diamonds$price)
diamonds$price %>% summary(object = .)
```

dplyr::filter()

- Filter (select) rows based on the condition of a column

```
diamonds %>% head
df.diamonds_ideal <- filter(diamonds, cut == "Ideal")
df.diamonds_ideal <- diamonds %>% filter(cut == "Ideal")
```

dplyr::select()

- Select columns from the dataset by names

```
df.diamonds_ideal %>% head
select(df.diamonds_ideal, carat, cut, color, price, clarity)
df.diamonds_ideal <- df.diamonds_ideal %>% select(., carat, cu
```

dplyr::mutate()

- Add columns to your dataset

```
df.diamonds_ideal %>% head
mutate(df.diamonds_ideal, price_per_carat = price/carat)
df.diamonds_ideal <- df.diamonds_ideal %>% mutate(price_per_ca
```

dplyr::arrange()

- Sort your data by columns

```
df.diamonds_ideal %>% head
arrange(df.diamonds_ideal, price)
df.diamonds_ideal %>% arrange(price, price_per_carat)
```

dplyr::summarize()

- Summarize columns by custom summary statistics

```
summarize(df.diamonds_ideal, length = n(), avg_price = mean(pr
df.diamonds_ideal %>% summarize(length = n(), avg_price = mean
```

dplyr::group_by()

- Summarize *subsets of* columns by custom summary statistics

```
group_by(diamonds, cut) %>% summarize(mean(price))
group_by(diamonds, cut, color) %>% summarize(mean(price))
```

The power of pipe %>%

- Summarize *subsets of* columns by custom summary statistics

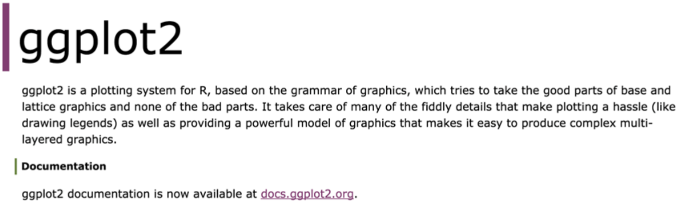
```
arrange(mutate(arrange(filter(tbl_df(diamonds), cut == "Ideal"
arrange(
  mutate(
    arrange(
      filter(tbl_df(diamonds), cut == "Ideal"),
    price),
  price_per_carat = price/carat),
price_per_carat)
diamonds %>% filter(cut == "Ideal") %>% arrange(price) %>% mut
```

ggplot2 - the grammar of graphics

ggplot2 package

http://ggplot2.org/

```
install.packages("ggplot2")
```



ggplot2

ggplot2 is a plotting system for R, based on the grammar of graphics, which tries to take the good parts of base and lattice graphics and none of the bad parts. It takes care of many of the fiddly details that make plotting a hassle (like drawing legends) as well as providing a powerful model of graphics that makes it easy to produce complex multi-layered graphics.

Documentation

ggplot2 documentation is now available at docs.ggplot2.org.

The basics of ggplot2 graphics

- Data mapped to graphical elements
- Add graphical layers and transformations
- Commands are chained with “+” sign

Object		Description
Data		The raw data that you want to plot
Aethetics	aes()	How to map your data on x, y axis, color, size, shape (aesthetics)
Geometries	geom_	The geometric shapes that will represent the data

data +
aesthetic mappings of data to plot coordinates +
geometry to represent the data

Examples of ggplot2 graphics

```
diamonds %>% filter(cut == "Good", color == "E") %>%
  ggplot(aes(x = price, y = carat)) +
  geom_point()  # aes(size = price) +
```

Try other geoms

```
  geom_smooth() # method = lm
  geom_line()
  geom_boxplot()
  geom_bar(stat="identity")
  geom_histogram()
```

Fine tuning ggplot2 graphics

Parameter		Description
Facets	facet_	Split one plot into multiple plots based on a grouping variable
Scales	scale_	Maps between the data ranges and the dimensions of the plot
Visual Themes	theme	The overall visual defaults of a plot: background, grids, axe, default typeface, sizes, colors, etc.
Statistical transformations	stat_	Statistical summaries of the data that can be plotted, such as quantiles, fitted curves (loess, linear models, etc.), sums etc.
Coordinate systems	coord_	Expressing coordinates in a system other than Cartesian

Putting it all together

```
diamonds %>%                 # Start with the 'diamonds' datas
  filter(cut == "Ideal") %>% # Then, filter rows where cut ==
  ggplot(aes(price)) +       # Then, plot using ggplot
  geom_histogram() +         # and plot histograms
  facet_wrap(~ color) +      # in a 'small multiple' plot, br
  ggtitle("Diamond price distribution per color") +
  labs(x="Price", y="Count") +
  theme(panel.background = element_rect(fill="lightblue")) +
  theme(plot.title = element_text(family="Trebuchet MS", size=
  theme(axis.title.y = element_text(angle=0)) +
  theme(panel.grid.minor = element_blank())
```

Other resources

- **Plotly** for R, `https://plot.ly/r/`
- **GoogleVis** for R, `https://cran.r-project.org/web/packages/googleVis/vignettes/googleVis_examples.html`
- **ggbio** - grammar of graphics for genomic data, `http://www.tengfei.name/ggbio/`