

Lab 02: BMI 5/625

Working in the Tidyverse

Alison Hill (w/ modifications by Steven Bedrick)



Tidyverse basics

Last week, we covered some basics:

- `←` (variable assignment)
- `%>%` (then...)
- `dplyr`, `ggplot2` (packages)
 - `install.packages("dplyr")` (1x per machine)
 - `library(dplyr)` (1x per work session)

Data for today

We'll use data from the Museum of Modern Art (MoMA)

- Publicly available on [GitHub](#)
- As analyzed by [fivethirtyeight.com](#)
- And by [others](#)

Get the data

Use this code chunk to import my cleaned CSV file:

```
library(readr)  
moma ← read_csv("../data/artworks-cleaned.csv")
```

Data wrangling:

All functions from `dplyr` package

A few basics:

- print a tibble
- `filter`
- `arrange`
- `mutate`

From Lab 01

- `glimpse`
- `distinct`
- `count`



Plus: %>%

image courtesy @LegoRLady

Three core functions: `filter`

`filter` subsets data according to a *predicate* (logical statement)

- Use for things like "remove subjects whose age is less than 18 years"

```
peds ← all.patients %>% filter(age ≤ 18)
```

- Note that predicates can be as complex as you like (examples to come)

Three core functions: arrange

`arrange` sorts a dataframe by one or more columns

```
peds ← peds %>% arrange(age)
```

- The default sort order is *ascending* (smallest to largest); you can reverse this in two ways:
- The `desc()` function, and negation:

```
# option 1:  
peds ← peds %>% arrange(desc(age))
```

```
# option 2:  
peds ← peds %>% arrange(-age)
```


Three core functions: mutate

`mutate` adds a new column (or replaces an existing one)

```
peds ← peds %>% mutate(age.in.months = age * 12)
```

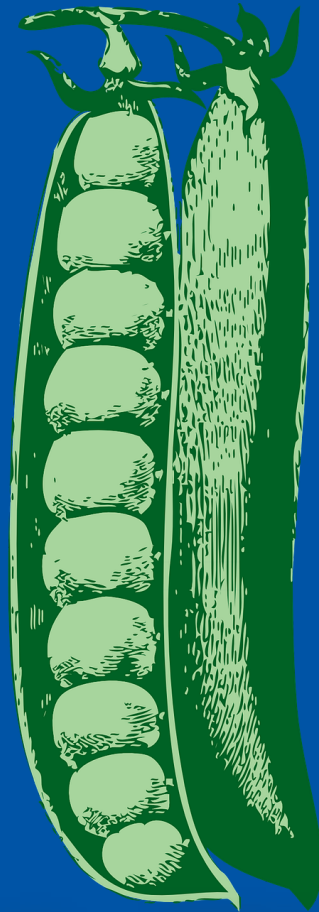
```
# convert to meters from feet  
peds ← peds %>% mutate(height = height * 0.305)
```

- Multiple columns can be worked on at the same time:

```
peds ← peds %>% mutate(  
  age.in.months = age * 12,  
  is.school.age = age ≥ 5,  
  height = height * 0.305  
)
```



Let's review some helpful functions for
filter



Base R + Tidyverse



First:

Logical Operators

Operator	Description	Usage
&	and	x & y
	or	x y
xor	exactly x or y	xor(x, y)
!	not	!x

Logical or (`|`) is inclusive, so `x | y` really means:

- x or
- y or
- both x & y

Exclusive or (`xor`) is exclusive, so `xor(x, y)` really means:

- x or
- y...
- but not both x & y

```
x ← c(0, 1, 0, 1)
y ← c(0, 0, 1, 1)
boolean_or ← x | y
exclusive_or ← xor(x, y)
cbind(x, y, boolean_or, exclusive_or)
```

	x	y	boolean_or	exclusive_or
[1,]	0	0	0	0
[2,]	1	0	1	1
[3,]	0	1	1	1
[4,]	1	1	1	0



Second:
Comparisons

?Comparison

Operator	Description	Usage
<	less than	<code>x < y</code>
<=	less than or equal to	<code>x <= y</code>
>	greater than	<code>x > y</code>
>=	greater than or equal to	<code>x >= y</code>
==	exactly equal to	<code>x == y</code>
!=	not equal to	<code>x != y</code>
%in%	group membership*	<code>x %in% y</code>
is.na	is missing	<code>is.na(x)</code>
!is.na	is not missing	<code>!is.na(x)</code>

*(shortcut to using `|` repeatedly with `==`)

Another level: `group_by`

New this week: `group_by`

Many `dplyr` verbs can be *grouped*

I.e., their operation can be performed on partitions of your data:

("average of `X`, *by* `Y`)

```
penguins %>% filter(!is.na(bill_length_mm)) %>%  
  group_by(species) %>%  
  summarise(mean_length=mean(bill_length_mm))
```

```
# A tibble: 3 × 2  
  species    mean_length  
  <fct>         <dbl>  
1 Adelie         38.8  
2 Chinstrap     48.8  
3 Gentoo        47.5
```

Most other `dplyr` verbs will "play nicely" with grouped data:

`arrange`, `slice`, `count`, `top_n`, etc.

Under the hood

What does `group_by` actually *do*?

```
penguins.grouped ← penguins %>% group_by(species)
penguins.grouped
```

```
# A tibble: 344 × 8
# Groups:   species [3]
  species island      bill_length_mm bill_depth_mm flipper_...1 body_...2 sex
  <fct>   <fct>          <dbl>          <dbl>         <int>     <int> <fct>
1 Adelie  Torgersen         39.1           18.7           181       3750 male
2 Adelie  Torgersen         39.5           17.4           186       3800 fema...
3 Adelie  Torgersen         40.3           18             195       3250 fema...
4 Adelie  Torgersen         NA             NA             NA         NA <NA>
5 Adelie  Torgersen         36.7           19.3           193       3450 fema...
6 Adelie  Torgersen         39.3           20.6           190       3650 male
7 Adelie  Torgersen         38.9           17.8           181       3625 fema...
8 Adelie  Torgersen         39.2           19.6           195       4675 male
9 Adelie  Torgersen         34.1           18.1           193       3475 <NA>
10 Adelie Torgersen         42             20.2           190       4250 <NA>
# ... with 334 more rows, and abbreviated variable names 'flipper_length_mm',
#   2body_mass_g
```

Multiple Groups

"How many males and females of each sex do we have?"

```
penguins %>% group_by(species, sex) %>% tally
```

Note that the resulting dataframe is still grouped by `species`!

```
penguins %>% group_by(species, sex)
```

```
# A tibble: 344 × 8
```

```
# Groups:   species, sex [8]
```

	species	island	bill_length_mm	bill_depth_mm	flipper_... ¹	body_... ²	sex
	<fct>	<fct>	<dbl>	<dbl>	<int>	<int>	<fct>
1	Adelie	Torgersen	39.1	18.7	181	3750	male
2	Adelie	Torgersen	39.5	17.4	186	3800	fema...
3	Adelie	Torgersen	40.3	18	195	3250	fema...
4	Adelie	Torgersen	NA	NA	NA	NA	<NA>
5	Adelie	Torgersen	36.7	19.3	193	3450	fema...
6	Adelie	Torgersen	39.3	20.6	190	3650	male
7	Adelie	Torgersen	38.9	17.8	181	3625	fema...
8	Adelie	Torgersen	39.2	19.6	195	4675	male
9	Adelie	Torgersen	34.1	18.1	193	3475	<NA>
10	Adelie	Torgersen	42	20.2	190	4250	<NA>

Lab 02: Challenge 1 (dplyr)

1. How many paintings (rows) are in `moma`? How many variables (columns) are in `moma`?
2. What is the first painting acquired by MoMA? Which year? Which artist? What title?
 - *Hint: you may want to look into `select + arrange`*
3. What is the oldest painting in the collection? Which year? Which artist? What title? (*see above hint*)
4. How many distinct artists are there?
5. Which artist has the most paintings in the collection? How many paintings are by this artist?
6. How many paintings are by male vs female artists?

If you want more:

1. How many artists of each gender are there?
2. In what year were the most paintings acquired? Created?
3. In what year was the first painting by a (solo) female artist acquired? When was that painting created? Which artist? What title?

From Last Week 2

From `ggplot2`:

- `aes(x = , y =)` (aesthetics)
- `aes(x = , y = , color =)` (add color)
- `aes(x = , y = , size =)` (add size)
- `+ facet_wrap(~)` (facetting)

"Old School" (Challenge 2)¹

- Sketch the graphics below on paper, where the `x`-axis is variable `year_created` and the `y`-axis is variable `year_acquired`

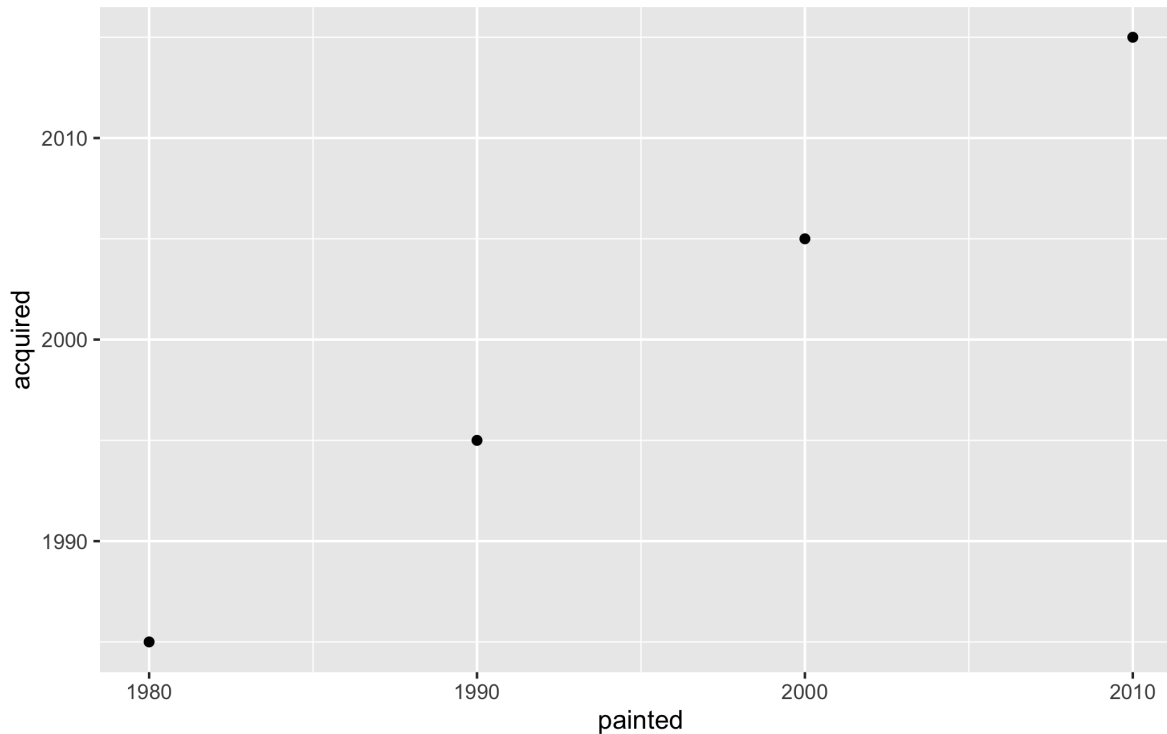
```
# A tibble: 4 × 4
  painted acquired  area gender
  <dbl>   <dbl> <dbl> <chr>
1   1980   1985     3 male
2   1990   1995     2 male
3   2000   2005     1 female
4   2010   2015     2 female
```

1. A scatter plot
2. A scatter plot where the `color` of the points corresponds to `gender`
3. A scatter plot where the `size` of the points corresponds to `area`
4. A version of (1), but with separate plots by gender

[1] Shamelessly borrowed with much appreciation to [Chester Ismay](#)

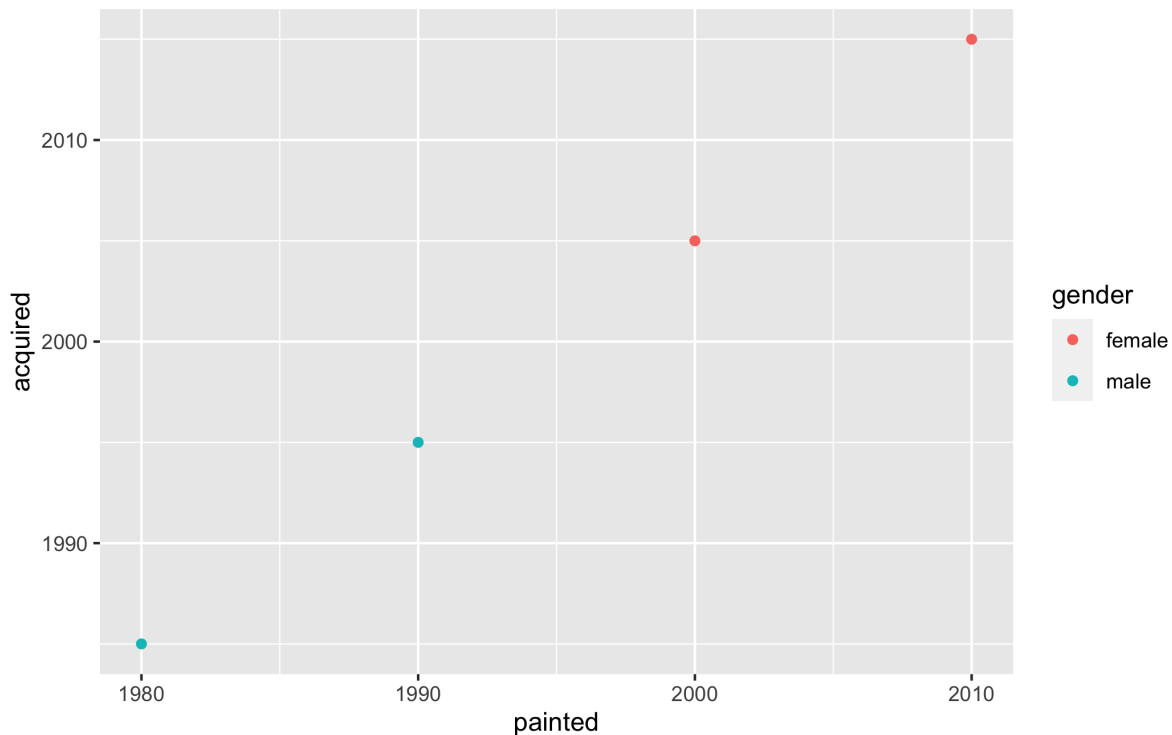
1. A scatterplot

```
library(ggplot2)  
ggplot(moma_ex, aes(painted, acquired)) +  
  geom_point()
```



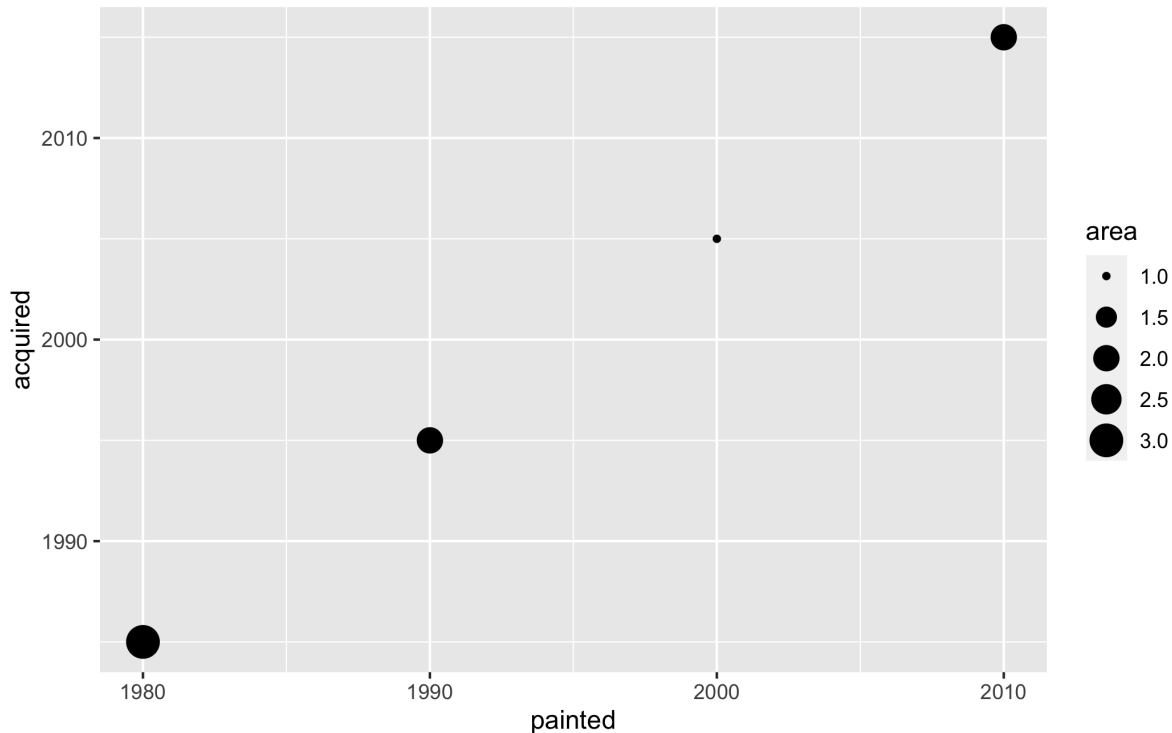
2. color points by gender

```
library(ggplot2)
ggplot(moma_ex, aes(painted, acquired, color = gender)) +
  geom_point()
```



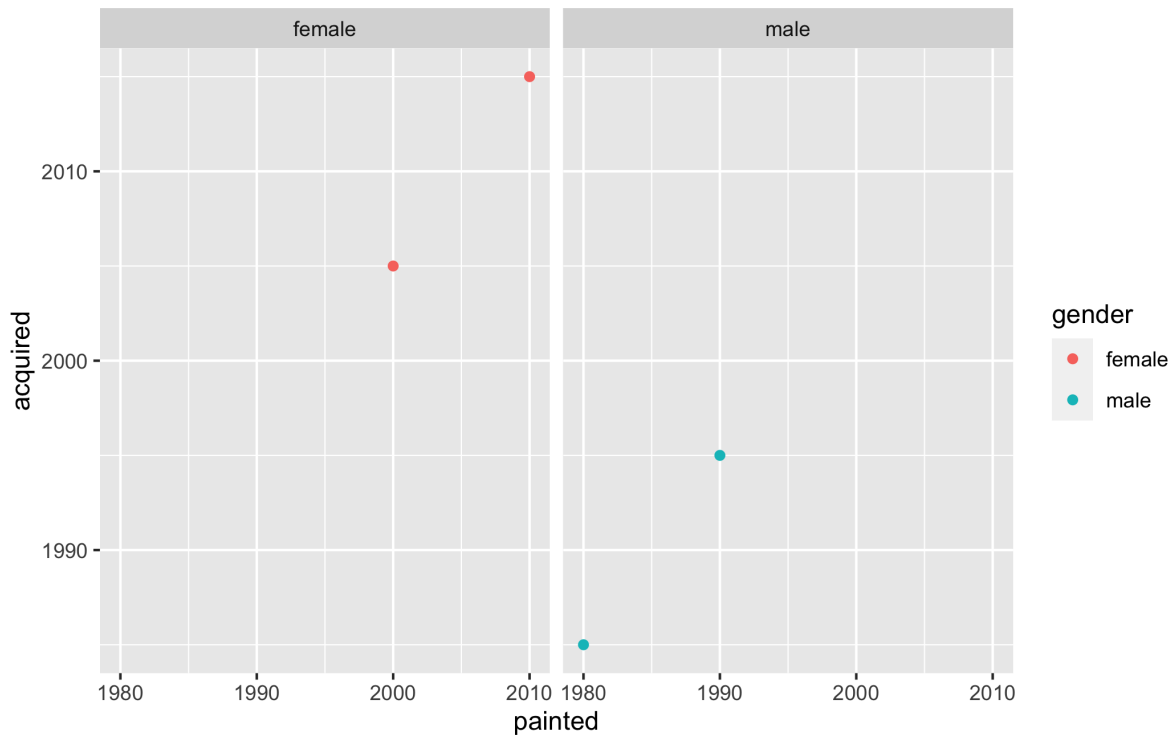
3. size points by area

```
library(ggplot2)
ggplot(moma_ex, aes(painted, acquired, size = area)) +
  geom_point()
```



4. Faceting

```
library(ggplot2)
ggplot(moma_ex, aes(painted, acquired, color = gender)) +
  geom_point() + facet_wrap(~gender)
```



The Five-Named Graphs

- Scatterplot: `geom_point()`
- Line graph: `geom_line()`
- Histogram: `geom_histogram()`
- Boxplot: `geom_boxplot()`
- Bar graph: `geom_bar()` or `geom_col` (see Lab 01)

Lab 02: Plotting Challenges

Challenges 3-5 are in the [Lab 02 code-through!](#)

<https://stevenbedrick.github.io/data-vis-labs-2023/02-moma.html>



Basics of ggplot2 and dplyr:

[R4DS ggplot2 chapter](#)

[ModernDive ggplot2 chapter](#)

[RStudio ggplot2 Cheatsheet](#)

[R4DS dplyr chapter](#)

[ModernDive dplyr chapter](#)

[RStudio dplyr Cheatsheet](#)