Introduction to Data Handling

Masatoshi Katabuchi

April 17, 2021@TokyoR91

Thanks to

- **Functional Programming** by Sara Altman, Bill Behrman and Hadley Wickham
  - https://github.com/dcl-docs/prog
- **Data Wrangling with R Workshop** by Emi Tanaka
  - https://github.com/emitanaka/biometrics2019
- **Tidyverse Workshop** by Emi Tanaka
  - https://github.com/emitanaka/datawrangle-workshop-ssavic

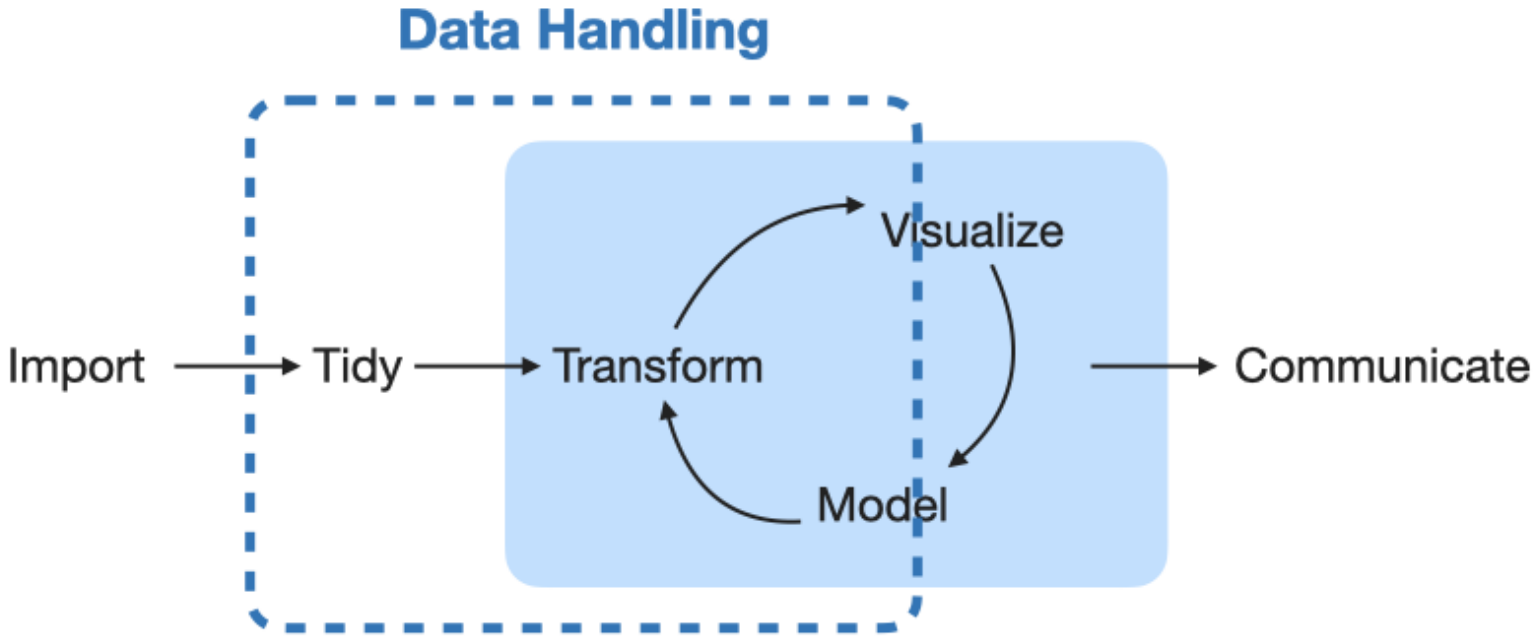
About me

- Masatoshi Katabuchi
- Plant Ecologist @ Xishuangbanna Tropical Botanical Garden
- Interests:
  - Data | Leaf | Beer

mattocci27@gmail.com | @mattocci | https://mattocci27.github.io

80/20 rule for R codes

~ 80% of your R code for data analysis and visualization will be spent cleaning and preparing data.

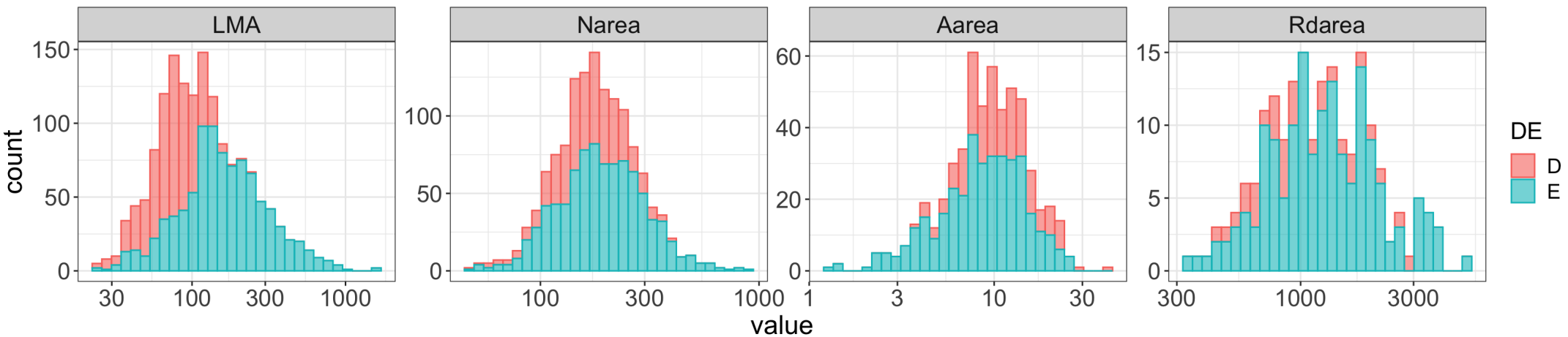


Hadley Wickham and Garrett Grolemund (2016) R for Data Science, O'Reilly Media, Inc.

Goal 1: using this data, make the plot below

```
glimpse(dat)
```

```
  Rows: 2,548
  Columns: 6
  $ sp_code <chr> "ADEFAS", "ARBMEN", "ARCTOM", "ARTCAL", "BACPIL", "CEACUN", "CEAOLI", "CERBE
  $ DE      <chr> "E", "E", "E", "D", "E", "E", "E", "E", "D", "D", "E", "E", "D", "D", "D", "
  $ LMA     <dbl> 281.838, 154.882, 141.254, 95.499, 107.152, 229.087, 120.226, 138.038, 74.13
  $ Nmass   <dbl> 1.172, 1.242, 1.033, NA, 2.443, 1.799, 2.128, 2.410, NA, 2.582, 1.667, 1.371
  $ Aarea   <dbl> 14.125, 11.220, 10.471, NA, 17.378, 22.909, 17.783, 24.547, NA, 7.762, 18.62
  $ Rdmass  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```



Wright et al. 2004. “The worldwide leaf economics spectrum” Nature 428:821–827

Goal 2: using this data, make the table below

```
glimpse(dat)
```

```
  Rows: 2,548
  Columns: 6
  $ sp_code <chr> "ADEFAS", "ARBMEN", "ARCTOM", "ARTCAL", "BACPIL", "CEACUN", "CEAOLI", "CERBE
  $ DE      <chr> "E", "E", "E", "D", "E", "E", "E", "E", "D", "D", "E", "E", "D", "D", "D", "
  $ LMA     <dbl> 281.838, 154.882, 141.254, 95.499, 107.152, 229.087, 120.226, 138.038, 74.13
  $ Nmass   <dbl> 1.172, 1.242, 1.033, NA, 2.443, 1.799, 2.128, 2.410, NA, 2.582, 1.667, 1.371
  $ Aarea   <dbl> 14.125, 11.220, 10.471, NA, 17.378, 22.909, 17.783, 24.547, NA, 7.762, 18.62
  $ Rdmass  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```

Summary stats

DE	mean_LMA	sd_LMA	mean_Aarea	sd_Aarea	n
D	78.1	27.6	11.4	5.2	602
E	199.7	154.6	9.7	4.9	1004
NA	85.1	60.0	13.8	6.8	942

Wright et al. 2004. “The worldwide leaf economics spectrum” Nature 428:821–827

Tidyverse

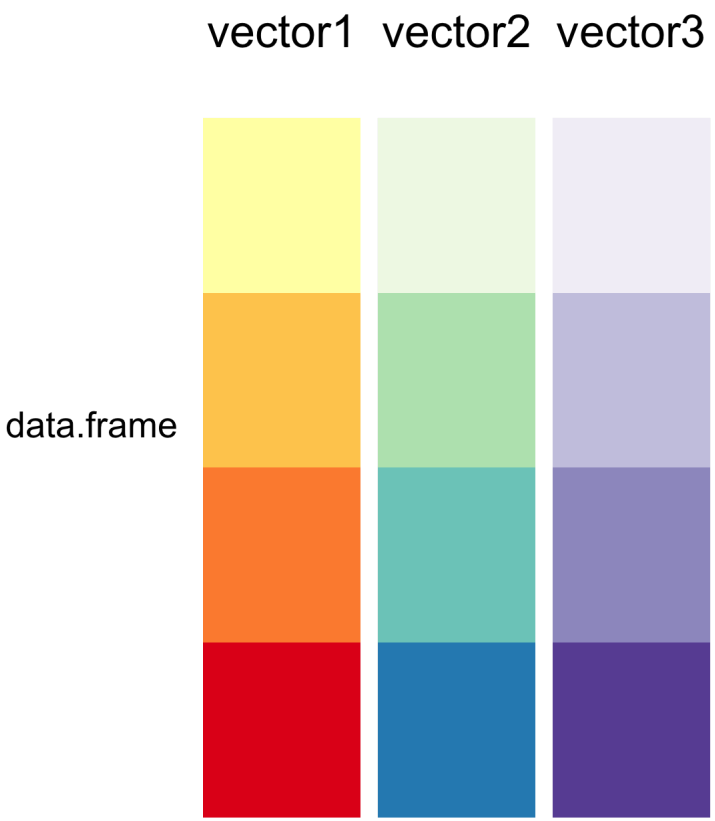
- **Tidyverse** refers to a collection of R-packages including `ggplot2`, `dplyr`, `tidyr`, `reader`, `purrr` ...
- Eight of these packages form the **core tidyverse**.

- `library(tidyverse)` is a short hand for `library(ggplot2)`, `library(dplyr)`, ..., `library(forcats)`
- We will use `dplyr` and `tidyr` for data handling today.

Wickham, H. et al. 2019. “Welcome to the Tidyverse.” Journal of Open. https://joss.theoj.org/papers/10.21105/joss.01686.

Data frames

- `data.frame` and `tibble` are lists of any types of vectors
- `matrix` can only contain a single type of vectors



mpg

```
# A tibble: 234 x 11
   manufacturer model      displ  year   cyl trans
   <chr>        <chr>      <dbl> <int> <int> <chr>
 1 audi         a4           1.8  1999     4 auto(
 2 audi         a4           1.8  1999     4 manua
 3 audi         a4           2    2008     4 manua
 4 audi         a4           2    2008     4 auto(
 5 audi         a4           2.8  1999     6 auto(
 6 audi         a4           2.8  1999     6 manua
 7 audi         a4           3.1  2008     6 auto(
 8 audi         a4 quattro   1.8  1999     4 manua
 9 audi         a4 quattro   1.8  1999     4 auto(
10 audi         a4 quattro   2    2008     4 manua
# … with 224 more rows
```

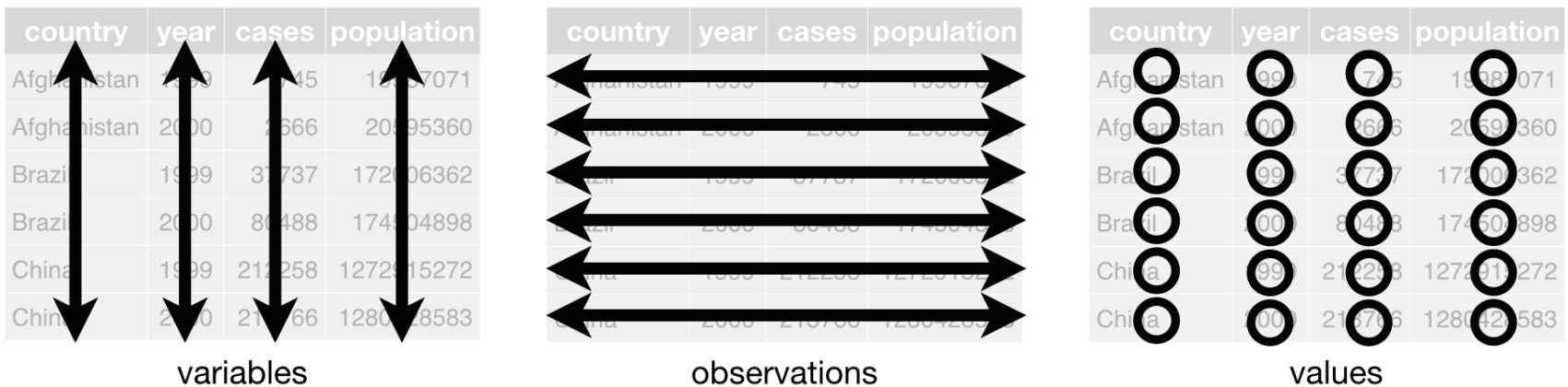
Tidy data

Typical aim of data handling is to make a tidy data

What is a tidy data?

- Each variable must have its own column
- Each observation must have its own row
- Each value must have its own cell

- easy to manipulate, model and visualize



Wickham, Hadley. 2014. “Tidy Data.” Journal of Statistical Software, Articles 59 (10): 1–23.

Data structure

Non-tidy data

person	treatment_a	treatment_b
John Smith	NA	2
Jane Doe	16	11
Mary Jhonson	3	1

Tidy data

person	treatment	result
John Smith	treatment_a	NA
Jane Doe	treatment_a	16
Mary Jhonson	treatment_a	3
John Smith	treatment_b	2
Jane Doe	treatment_b	11
Mary Jhonson	treatment_b	1

Wickham, Hadley. 2014. “Tidy Data.” Journal of Statistical Software, Articles 59 (10): 1–23.

```
non_tidy1
```

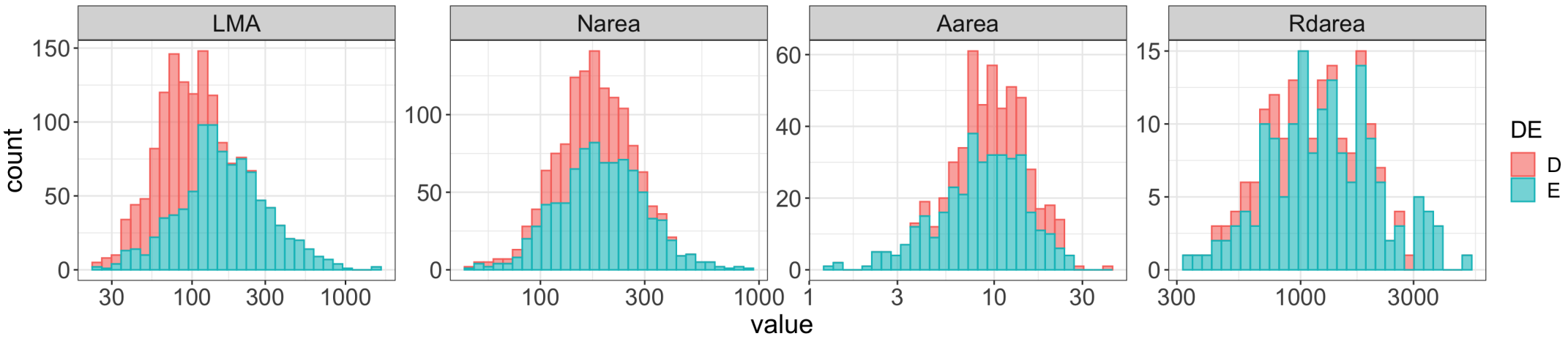
```
# A tibble: 3 x 3
  person       treatment_a treatment_b
  <chr>              <dbl>       <dbl>
1 John Smith            NA           2
2 Jane Doe              16          11
3 Mary Jhonson           3           1
```

```
non_tidy1

  # A tibble: 3 x 3
    person       treatment_a treatment_b
    <chr>              <dbl>       <dbl>
  1 John Smith            NA           2
  2 Jane Doe              16          11
  3 Mary Jhonson           3           1

non_tidy1 %>%
  pivot_longer(2:3, names_to = "treatment", values_to = "result") %>%
  arrange(treatment)

  # A tibble: 6 x 3
    person       treatment   result
    <chr>        <chr>        <dbl>
  1 John Smith   treatment_a     NA
  2 Jane Doe     treatment_a     16
  3 Mary Jhonson treatment_a      3
  4 John Smith   treatment_b      2
  5 Jane Doe     treatment_b     11
  6 Mary Jhonson treatment_b      1
```

Goal 1: using this data, make the plot below

```
glimpse(dat)
```

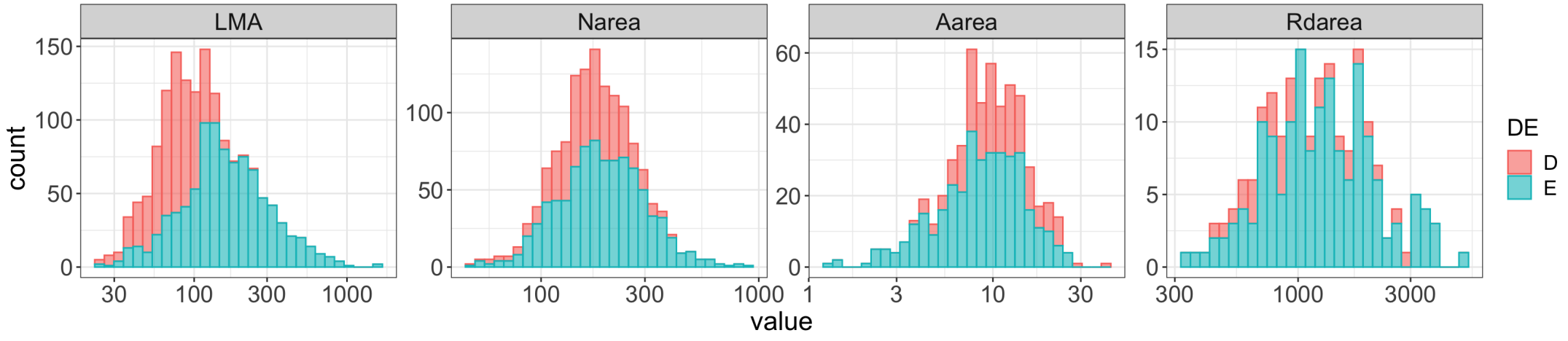
```
Rows: 2,548
Columns: 6
$ sp_code <chr> "ADEFAS", "ARBMEN", "ARCTOM", "ARTCAL", "BACPIL", "CEACUN", "CEAOLI", "CERBE
$ DE      <chr> "E", "E", "E", "D", "E", "E", "E", "E", "D", "D", "E", "E", "D", "D", "D", "
$ LMA     <dbl> 281.838, 154.882, 141.254, 95.499, 107.152, 229.087, 120.226, 138.038, 74.13
$ Nmass   <dbl> 1.172, 1.242, 1.033, NA, 2.443, 1.799, 2.128, 2.410, NA, 2.582, 1.667, 1.371
$ Aarea   <dbl> 14.125, 11.220, 10.471, NA, 17.378, 22.909, 17.783, 24.547, NA, 7.762, 18.62
$ Rdmass  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```



Goal 1: using this data, make the plot below

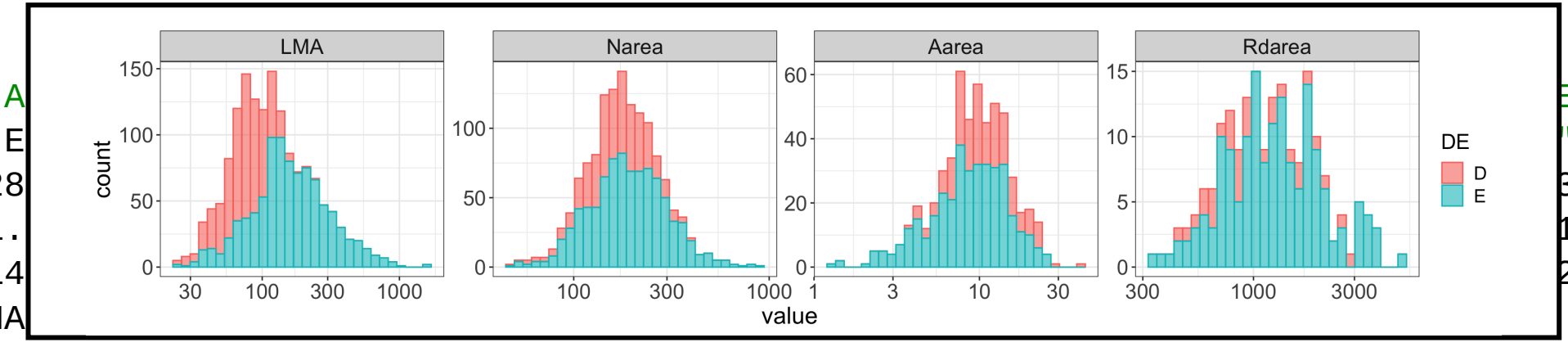
```
glimpse(

Rows: 2,
Columns:
$ sp_coc
$ DE
$ LMA
$ Nmass
$ Aarea
$ Rdmass
```

- DE: deciduous, evergreen, NA -> remove missing values
- LMA: leaf mass / leaf area
- Nmass: leaf nitrogen / leaf mass -> Narea = Nmass × LMA
- Aarea: photosynthetic rates / leaf area
- Rdmass: respiration rates / leaf mass -> Rdarea = Rdmass × LMA



Mapping variable to aesthetistic (see `ggplot2`)

```
Rows: 2,548
Columns: 6
$ sp_code <chr> "A
$ DE       <chr> "E
$ LMA      <dbl> 28
$ Nmass    <dbl> 1.
$ Aarea    <dbl> 14
$ Rdmass   <dbl> NA
```




```
ggplot(<DATA>, aes(x = <VAR>)) +
  geom_histogram() +
  facet_wrap(~ <GROUP>,
             scale = "free",
             nrow = 1)
```

🔧 Data wrangling

The data we *have*

LMA	Narea	Aarea	Rdarea
1	1	1	1
2	2	2	2

The data we *need*

name	value
LMA	1
LMA	2
Narea	1
Narea	2
Aarea	1
Aarea	2
Rdarea	1
Rdarea	2

🔧 Data wrangling

The data we *have*

LMA	Narea	Aarea	Rdarea
1	1	1	1
2	2	2	2

The data we *need*

name	value
LMA	1
LMA	2
Narea	1
Narea	2
Aarea	1
Aarea	2
Rdarea	1
Rdarea	2

```
ggplot(<DATA>, aes(x = value)) +
  geom_histogram() +
  facet_wrap(~ name,
             scale = "free",
             nrow = 1)
```

🔧 Data wrangling using `tidyr::pivot_longer`

The following commands all produce the same output on the right:

```
pivot_longer(dat,
  cols = c("LMA", "Nmass", "Aarea", "Rdmass"))

pivot_longer(dat,
  cols = c(LMA, Nmass, Aarea, Rdmass))

pivot_longer(dat, cols = LMA:Rdmass)

pivot_longer(dat, cols = 2:5)
```

```
dat %>%
  pivot_longer(cols = LMA:Rdmass)


# A tibble: 10,192 x 4
   sp_code DE    name    value
   <chr>   <chr> <chr>   <dbl>
 1 ADEFAS  E     LMA    282.
 2 ADEFAS  E     Nmass    1.17
 3 ADEFAS  E     Aarea   14.1
 4 ADEFAS  E     Rdmass  NA
 5 ARBMEN  E     LMA    155.
 6 ARBMEN  E     Nmass    1.24
 7 ARBMEN  E     Aarea   11.2
 8 ARBMEN  E     Rdmass  NA
 9 ARCTOM  E     LMA    141.
10 ARCTOM  E     Nmass    1.03
# … with 10,182 more rows
```

⋮⋮

pipes %>%

- `f(<data>, <argA>) = <data> %>% f(<argA>)`
- `g(f(<data>, <argA>), <argB>) = f(<data>, <argA>) %>% g(<argB>)`

pipes %>%

- `f(<data>, <argA>) = <data> %>% f(<argA>)`
- `g(f(<data>, <argA>), <argB>) = f(<data>, <argA>) %>% g(<argB>)`

Let's say you want to apply function F to x first, then apply G, then apply H, then apply I, then K ...

pipes %>%

- `f(<data>, <argA>) = <data> %>% f(<argA>)`
- `g(f(<data>, <argA>), <argB>) = f(<data>, <argA>) %>% g(<argB>)`

Let's say you want to apply function `F` to `x` first, then apply `G`, then apply `H`, then apply `I`, then `K` ...

Which one do you prefer?

`K(I(H(G(F(x)))))` or `F(x) %>% G %>% H %>% I %>% K`

Filter observations using `dplyr::filter`

```
dat
```

```
# A tibble: 2,548 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 2,538 more rows
```

```
dat$DE %>% unique
```

```
[1] "E" "D" NA
```

Filter observations using `dplyr::filter`

```
dat
```

```
# A tibble: 2,548 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 2,538 more rows
```

```
dat$DE %>% unique
```

```
[1] "E" "D" NA
```

```
dat2 <- dat %>% filter(!is.na(DE))
dat2
```

```
# A tibble: 1,606 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 1,596 more rows
```

```
dat2$DE %>% unique
```

```
[1] "E" "D"
```

Filter observations using `dplyr::filter`

```
dat
```

```
# A tibble: 2,548 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 2,538 more rows
```

```
dat$DE %>% unique
```

```
[1] "E" "D" NA
```

Base R

```
dat[!is.na(dat$DE), ]
```

```
dat2 <- dat %>% filter(!is.na(DE))
dat2
```

```
# A tibble: 1,606 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 1,596 more rows
```

```
dat2$DE %>% unique
```

```
[1] "E" "D"
```

Filter observations using `dplyr::filter`

```
dat
```

```
# A tibble: 2,548 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 2,538 more rows
```

```
dat$DE %>% unique
```

```
[1] "E" "D" NA
```

Base R

```
dat[!is.na(dat$DE), ]
```

Base R

```
subset(dat, !is.na(dat$DE))
```

```
dat2 <- dat %>% filter(!is.na(DE))
dat2
```

```
# A tibble: 1,606 x 6
   sp_code DE      LMA Nmass Aarea Rdmass
   <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
 1 ADEFAS  E     282.   1.17 14.1      NA
 2 ARBMEN  E     155.   1.24 11.2      NA
 3 ARCTOM  E     141.   1.03 10.5      NA
 4 ARTCAL  D      95.5 NA    NA        NA
 5 BACPIL  E     107.   2.44 17.4      NA
 6 CEACUN  E     229.   1.80 22.9      NA
 7 CEAOLI  E     120.   2.13 17.8      NA
 8 CERBET  E     138.   2.41 24.5      NA
 9 CLELAS  D      74.1 NA    NA        NA
10 DIROCC  D      56.2  2.58  7.76     NA
# … with 1,596 more rows
```

```
dat2$DE %>% unique
```

```
[1] "E" "D"
```

Make new variables using `dplyr::mutate`

```
dat %>% head(3)
```

```
  # A tibble: 3 x 6
    sp_code DE      LMA Nmass Aarea Rdmass
    <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
  1 ADEFAS  E      282.  1.17  14.1     NA
  2 ARBMEN  E      155.  1.24  11.2     NA
  3 ARCTOM  E      141.  1.03  10.5     NA
```

Make new variables using dplyr::mutate

```
dat %>% head(3)
```

```
# A tibble: 3 x 6
  sp_code DE      LMA Nmass Aarea Rdmass
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA
2 ARBMEN  E      155.  1.24  11.2     NA
3 ARCTOM  E      141.  1.03  10.5     NA
```

```
dat %>%
  mutate(Narea = LMA * Nmass) %>%
  head(3)
```

```
# A tibble: 3 x 7
  sp_code DE      LMA Nmass Aarea Rdmass Narea
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl> <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA  330.
2 ARBMEN  E      155.  1.24  11.2     NA  192.
3 ARCTOM  E      141.  1.03  10.5     NA  146.
```

Make new variables using `dplyr::mutate`

```
dat %>% head(3)
```

```
# A tibble: 3 x 6
  sp_code DE       LMA Nmass Aarea Rdmass
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA
2 ARBMEN  E      155.  1.24  11.2     NA
3 ARCTOM  E      141.  1.03  10.5     NA
```

```
dat %>%
  mutate(Narea = LMA * Nmass) %>%
  head(3)
```

Base R

```
dat$Narea <- dat$LMA * dat$Nmass
```

```
# A tibble: 3 x 7
  sp_code DE       LMA Nmass Aarea Rdmass Narea
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl> <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA  330.
2 ARBMEN  E      155.  1.24  11.2     NA  192.
3 ARCTOM  E      141.  1.03  10.5     NA  146.
```

Select variables using `dplyr::select`

```
dat %>% head(3)
```

```
  # A tibble: 3 x 6
    sp_code DE      LMA Nmass Aarea Rdmass
    <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
  1 ADEFAS  E      282.  1.17  14.1     NA
  2 ARBMEN  E      155.  1.24  11.2     NA
  3 ARCTOM  E      141.  1.03  10.5     NA
```

Select variables using `dplyr::select`

```
dat %>% head(3)
```

```
# A tibble: 3 x 6
  sp_code DE      LMA Nmass Aarea Rdmass
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA
2 ARBMEN  E      155.  1.24  11.2     NA
3 ARCTOM  E      141.  1.03  10.5     NA
```

```
dat %>%
  dplyr::select(-Nmass, -Rdmass) %>%
  head(3)
```

```
# A tibble: 3 x 4
  sp_code DE      LMA Aarea
  <chr>   <chr> <dbl> <dbl>
1 ADEFAS  E      282.  14.1
2 ARBMEN  E      155.  11.2
3 ARCTOM  E      141.  10.5
```

:::

Select variables using dplyr::select

```
dat %>% head(3)
```

```
# A tibble: 3 x 6
  sp_code DE      LMA Nmass Aarea Rdmass
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA
2 ARBMEN  E      155.  1.24  11.2     NA
3 ARCTOM  E      141.  1.03  10.5     NA
```

```
dat %>%
  dplyr::select(-Nmass, -Rdmass) %>%
  head(3)
```

```
# A tibble: 3 x 4
  sp_code DE      LMA Aarea
  <chr>   <chr> <dbl> <dbl>
1 ADEFAS  E      282.  14.1
2 ARBMEN  E      155.  11.2
3 ARCTOM  E      141.  10.5
```

```
dat %>%
  dplyr::select(sp_code, DE, LMA, Aarea)
```

:::

Select variables using `dplyr::select`

```
dat %>% head(3)
```

```
# A tibble: 3 x 6
  sp_code DE      LMA Nmass Aarea Rdmass
  <chr>   <chr> <dbl> <dbl> <dbl>  <dbl>
1 ADEFAS  E      282.  1.17  14.1     NA
2 ARBMEN  E      155.  1.24  11.2     NA
3 ARCTOM  E      141.  1.03  10.5     NA
```

```
dat %>%
  dplyr::select(-Nmass, -Rdmass) %>%
  head(3)
```

```
# A tibble: 3 x 4
  sp_code DE      LMA Aarea
  <chr>   <chr> <dbl> <dbl>
1 ADEFAS  E      282.  14.1
2 ARBMEN  E      155.  11.2
3 ARCTOM  E      141.  10.5
```

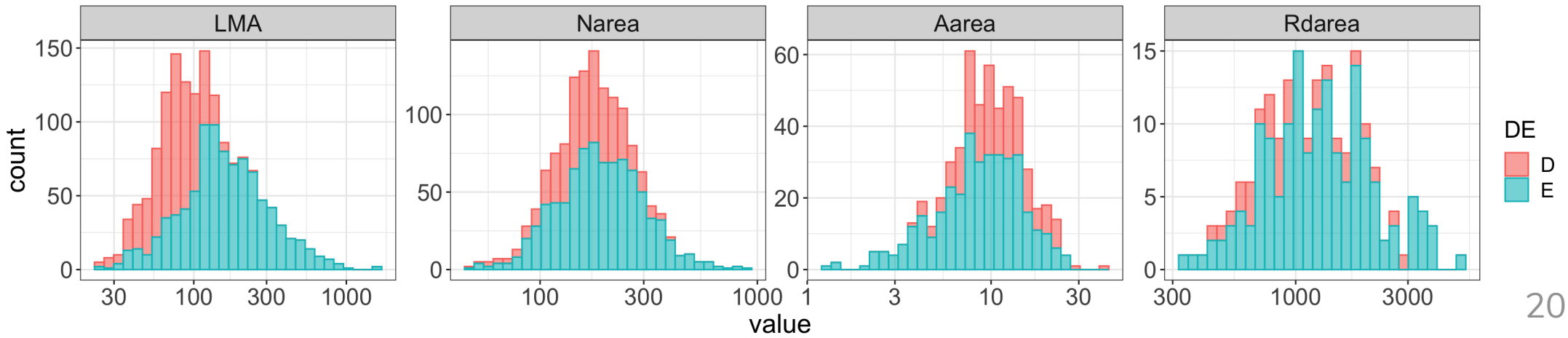
```
dat %>%
  dplyr::select(sp_code, DE, LMA, Aarea)
```

Base R

```
dat[, -which(names(dat) == "Nmass" | names(dat) == "Rdmass")]
```

Goal 1: Data wrangling for visualization

```
dat %>%
  filter(!is.na(DE)) %>%
  mutate(Narea = LMA * Nmass) %>%
  mutate(Rdarea = LMA * Rdmass) %>%
  dplyr::select(-Nmass, -Rdmass) %>%
  pivot_longer(cols = LMA:Rdarea) %>%
  mutate(name = factor(name, levels = c("LMA", "Narea", "Aarea", "Rdarea"))) %>%
  ggplot(., aes(x = value, fill = DE)) +
  geom_histogram(alpha = 0.6, aes(col = DE)) +
  facet_wrap(~ name, scale = "free", nrow = 1) +
  scale_x_log10() +
  theme(strip.text = element_text(size = 16))
```



Goal 2: using this data, make the table below

```
glimpse(dat)
```

```
  Rows: 2,548
  Columns: 6
  $ sp_code <chr> "ADEFAS", "ARBMEN", "ARCTOM", "ARTCAL", "BACPIL", "CEACUN", "CEAOLI", "CERBE
  $ DE      <chr> "E", "E", "E", "D", "E", "E", "E", "E", "D", "D", "E", "E", "D", "D", "D", "
  $ LMA     <dbl> 281.838, 154.882, 141.254, 95.499, 107.152, 229.087, 120.226, 138.038, 74.13
  $ Nmass   <dbl> 1.172, 1.242, 1.033, NA, 2.443, 1.799, 2.128, 2.410, NA, 2.582, 1.667, 1.371
  $ Aarea   <dbl> 14.125, 11.220, 10.471, NA, 17.378, 22.909, 17.783, 24.547, NA, 7.762, 18.62
  $ Rdmass  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
```

Summary stats

DE	mean_LMA	sd_LMA	mean_Aarea	sd_Aarea	n
D	78.1	27.6	11.4	5.2	602
E	199.7	154.6	9.7	4.9	1004
NA	85.1	60.0	13.8	6.8	942

Calculating summary stats using `dplyr::group_by`

- Calculate the mean of LMA values for each DE (group)

```
dat %>%
  group_by(DE) %>%
  summarise(mean_LMA = mean(LMA, na.rm = TRUE))


 # A tibble: 3 x 2
   DE    mean_LMA
   <chr>    <dbl>
 1 D         78.1
 2 E        200.
 3 <NA>      85.1
```

Calculating summary stats using `dplyr::group_by`

- Calculate the mean of LMA values for each DE (group)

```
dat %>%
  group_by(DE) %>%
  summarise(mean_LMA = mean(LMA, na.rm = TRUE))


# A tibble: 3 x 2
  DE    mean_LMA
  <chr>    <dbl>
1 D         78.1
2 E        200.
3 <NA>      85.1
```

- Calculate the mean of LMA values and count the sample size for each DE (group)

```
dat %>%
  group_by(DE) %>%
  summarise(
    mean_LMA = mean(LMA, na.rm = TRUE),
    n = n())


# A tibble: 3 x 3
  DE    mean_LMA     n
  <chr>    <dbl> <int>
1 D         78.1   602
2 E        200.   1004
3 <NA>      85.1   942
```

Goal 2: Data wrangling for summary stats

```
dat %>%
  group_by(DE) %>%
  summarise(mean_LMA = mean(LMA, na.rm = TRUE) ,
            sd_LMA = sd(LMA, na.rm = TRUE),
            mean_Aarea = mean(Aarea, na.rm = TRUE),
            sd_Aarea = sd(Aarea, na.rm = TRUE),
            n = n())
```

```
# A tibble: 3 x 6
  DE    mean_LMA sd_LMA mean_Aarea sd_Aarea     n
  <chr>    <dbl>  <dbl>      <dbl>    <dbl> <int>
1 D         78.1   27.6      11.4      5.21   602
2 E        200.   155.        9.72     4.85  1004
3 <NA>      85.1   60.0      13.8      6.80   942
```

Cheat sheet

Data Transformation with dplyr : : CHEAT SHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:

Each **variable** is in its own **column**

&

Each **observation**, or **case**, is in its own **row**

pipes

x %>% f(y) becomes **f(x, y)**

Summarise Cases

These apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

summary function

summarise(.data, …)
Compute table of summaries.
summarise(mtcars, avg = mean(mpg))

count(x, ..., wt = NULL, sort = FALSE)
Count number of rows in each group defined by the variables in … Also **tally**().
count(iris, Species)

VARIATIONS

summarise_all() - Apply funs to every column.
summarise_at() - Apply funs to specific columns.
summarise_if() - Apply funs to all cols of one type.

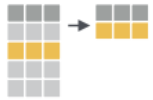
Group Cases

Use **group_by()** to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and

Manipulate Cases

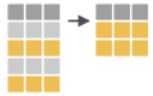
EXTRACT CASES

Row functions return a subset of rows as a new table.



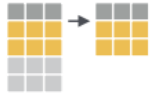
filter(.data, …) Extract rows that meet logical criteria. *filter(iris, Sepal.Length > 7)*

distinct(.data, ..., .keep_all = FALSE) Remove rows with duplicate values.
distinct(iris, Species)



sample_frac(tbl, size = 1, replace = FALSE, weight = NULL, .env = parent.frame()) Randomly select fraction of rows.
sample_frac(iris, 0.5, replace = TRUE)

sample_n(tbl, size, replace = FALSE, weight = NULL, .env = parent.frame()) Randomly select size rows. *sample_n(iris, 10, replace = TRUE)*



slice(.data, …) Select rows by position.
slice(iris, 10:15)

top_n(x, n, wt) Select and order top n entries (by group if grouped data). *top_n(iris, 5, Sepal.Width)*

Logical and boolean operators to use with filter()

<	<=	is.na()	%in%	|	xor()
>	>=	!is.na()	!	&	

See **?base::Logic** and **?Comparison** for help.

Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.

pull(.data, var = -1) Extract column values as a vector. Choose by name or index.
pull(iris, Sepal.Length)

select(.data, …)
Extract columns as a table. Also **select_if()**.
select(iris, Sepal.Length, Species)

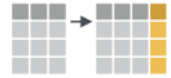
Use these helpers with select (),
e.g. select(iris, starts_with("Sepal"))

contains(match)	**num_range**(prefix, range)	**:**, e.g. mpg:cyl
ends_with(match)	**one_of**(…)	**-**, e.g, -Species
matches(match)	**starts_with**(match)	

MAKE NEW VARIABLES

These apply **vectorized functions** to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).

vectorized function



mutate(.data, …)
Compute new column(s).
mutate(mtcars, gpm = 1/mpg)

transmute(.data, …)
Compute new column(s), drop others.
transmute(mtcars, gpm = 1/mpg)

mutate_all(.tbl, .funs, …) Apply funs to every