Restructuring Data

Simon Andrews

V2020-07

Babraham Bioinformatics

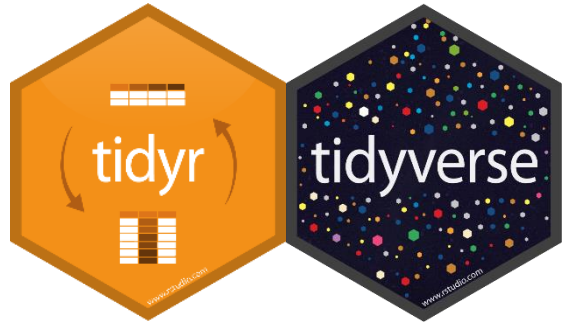
Wide Format

Gene	WT_1	WT_2	WT_3	KO_1	KO_2	KO_3
ABC1	8.86	4.18	8.90	4.00	14.52	13.39
DEF1	29.60	41.22	36.15	11.18	16.68	1.64

- Compact
- Easy to read
- Shows linkage for genes

- No explicit genotype or replicate
- Values spread out over multiple rows and columns
- Not extensible to more metadata

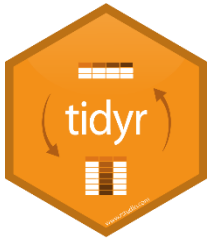
Long Format



Gene	Genotype	Replicate	Value
ABC1	WT	1	8.86
ABC1	WT	2	4.18
ABC1	WT	3	8.90
ABC1	KO	1	4.00
ABC1	KO	2	14.52
ABC1	KO	3	13.39
DEF1	WT	1	29.60
DEF1	WT	2	41.22
DEF1	WT	3	36.15
DEF1	KO	1	11.18
DEF1	KO	2	16.68
DEF1	KO	3	1.64

- More verbose (repeated values)
- Explicit genotype and replicate
- All values in a single column
- Extensible to more metadata

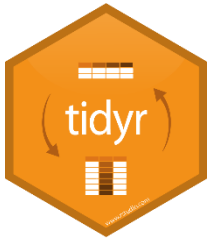
Converting to "Tidy" format




```
# A tibble: 3 x 8
  Gene    Chr   Start     End  WT_1  WT_2  KO_1  KO_2
  <chr> <dbl>   <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>
1 Gnai3     2  163898  167465  9.39  10.9  33.5  81.9
2 Pbsn      5 4888573 4891351 91.7   59.6  45.3  82.3
3 Cdc45     7 1250084 1262669 69.2   36.1  54.4  38.1
```

- Put all measures into a single column
- Add a 'genotype' and 'replicate' column
- Duplicate the gene information as required

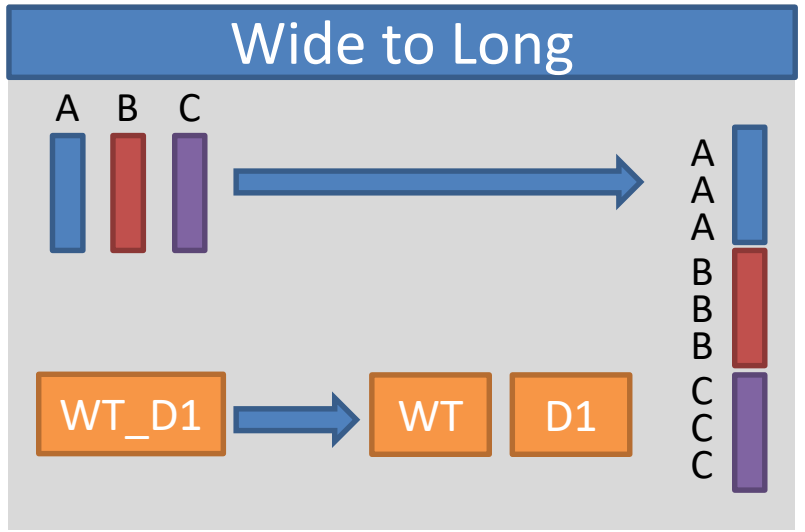
Converting to "Tidy" format

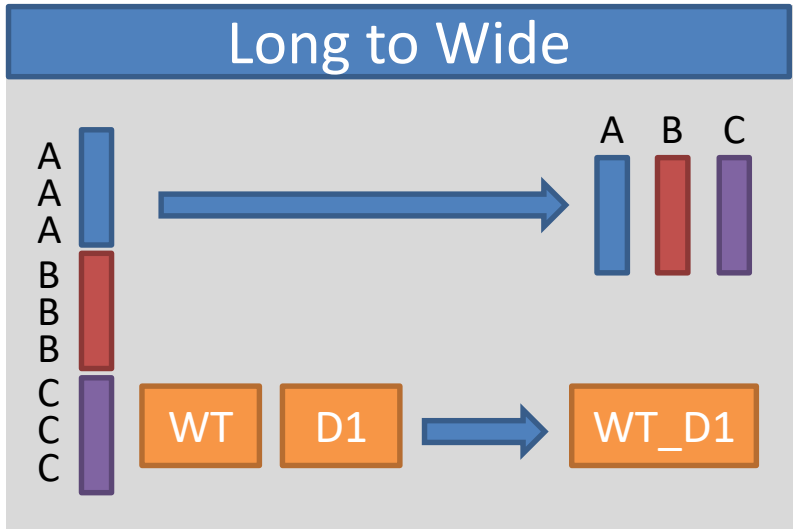



```
# A tibble: 12 x 7
   Gene    Chr   Start     End genotype replicate value
   <chr> <dbl>   <dbl>   <dbl> <chr>         <int> <dbl>
 1 Gnai3     2  163898  167465 WT                1  9.39
 2 Pbsn      5 4888573 4891351 WT                1 91.7
 3 Cdc45     7 1250084 1262669 WT                1 69.2
 4 Gnai3     2  163898  167465 WT                2 10.9
 5 Pbsn      5 4888573 4891351 WT                2 59.6
 6 Cdc45     7 1250084 1262669 WT                2 36.1
 7 Gnai3     2  163898  167465 KO                1 33.5
 8 Pbsn      5 4888573 4891351 KO                1 45.3
 9 Cdc45     7 1250084 1262669 KO                1 54.4
10 Gnai3     2  163898  167465 KO                2 81.9
11 Pbsn      5 4888573 4891351 KO                2 82.3
12 Cdc45     7 1250084 1262669 KO                2 38.1
```

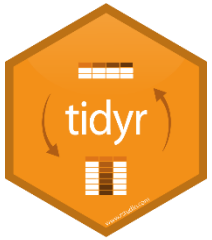
Tidying operations

- `pivot_longer`
  - Takes multiple columns of the same type and puts them into a pair of key-value columns
- `separate`
  - Splits a delimited column into multiple columns
- `pivot_wider`
  - Takes a key-value column pair and spreads them out to multiple columns of the same type
- `unite`
  - Combines multiple columns into one





Converting to "Tidy" format



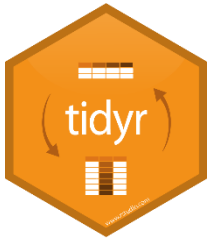

```
non.normalised %>%

  pivot_longer(
    cols=WT_1:KO_2,
    names_to="sample",
    values_to="value“
  ) %>%

  separate(
    col=sample,
    into=c("genotype","replicate"),
    convert = TRUE,
    sep="_"
  )
```

```
# A tibble: 3 x 8
  Gene    Chr   Start     End  WT_1  WT_2  KO_1  KO_2
  <chr> <dbl>   <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>
1 Gnai3     2  163898  167465  9.39  10.9  33.5  81.9
2 Pbsn      5 4888573 4891351 91.7   59.6  45.3  82.3
3 Cdc45     7 1250084 1262669 69.2   36.1  54.4  38.1
```

```
# A tibble: 12 x 7
   Gene    Chr   Start     End genotype replicate value
   <chr> <dbl>   <dbl>   <dbl> <chr>         <int> <dbl>
 1 Gnai3     2  163898  167465 WT                1  9.39
 2 Pbsn      5 4888573 4891351 WT                1 91.7
 3 Cdc45     7 1250084 1262669 WT                1 69.2
```

convert=TRUE makes separate re-detect the type of the column, so replicate becomes a numeric value

Pivoting Examples



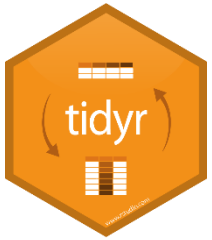

```
> pivot.data
# A tibble: 4 x 3
  gene     WT    KO
  <chr> <dbl> <dbl>
1 ABC1  18608  7831
2 DEF1  31988 55502
3 GHI1   7647 93299
4 JKL1  96002 47945
```

- Log transform all of the values
- Pivot longer
  - Which columns are we pivoting?
  - What do we want to call the new column of names?
  - What do we want to call the new column of values?

```
pivot.data %>%
  pivot_longer(
    cols=WT:KO,
    names_to = "Condition",
    values_to = "Count"
  ) -> pivot.long
```

```
# A tibble: 8 x 3
  gene  Condition Count
  <chr> <chr>     <dbl>
1 ABC1  WT        18608
2 ABC1  KO         7831
3 DEF1  WT        31988
4 DEF1  KO        55502
5 GHI1  WT         7647
6 GHI1  KO        93299
7 JKL1  WT        96002
8 JKL1  KO        47945
```

Pivoting Examples




```
> pivot.long
# A tibble: 8 x 3
  gene  Condition Count
  <chr> <chr>     <dbl>
1 ABC1  WT         14.2
2 ABC1  KO         12.9
3 DEF1  WT         15.0
4 DEF1  KO         15.8
5 GHI1  WT         12.9
6 GHI1  KO         16.5
7 JKL1  WT         16.6
8 JKL1  KO         15.5
```

- Plot WT vs KO
- Pivot wider
  - Which column of names?
  - Which column of values?

```
pivot.long %>%
  pivot_wider(
    names_from = Condition,
    values_from = Count
  )
```

```
# A tibble: 4 x 3
  gene     WT    KO
  <chr> <dbl> <dbl>
1 ABC1   14.2  12.9
2 DEF1   15.0  15.8
3 GHI1   12.9  16.5
4 JKL1   16.6  15.5
```

Exercise
Restructuring data into ‘tidy’ format

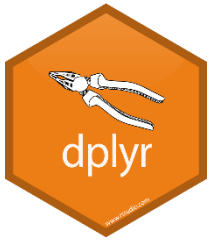
We have provided 3 files which need to be restructured into tidy format. These all need turning from wide format to long format so use a combination of `pivot_wider` (and if needed, `separate`) to restructure them.

If you have time – take the restructured data and use pivot_wider (and if needed, unite) to put them back into wide format

Babraham Bioinformatics

Grouping and Summarising

Grouping and Summarising Workflow



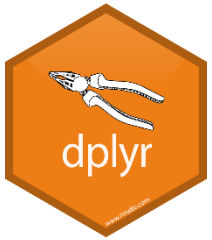
1. Load a tibble with repeated values in one or more columns

2. Use `group_by` to select all of the categorical columns you want to combine to define your groups

3. Run `summarise` saying how you want to combine the quantitative values

4. Run `ungroup` to remove any remaining group information

Grouping and Summarising Workflow



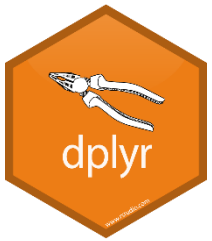
1. Load a tibble with repeated values in one or more columns

2. Use `group_by` to select all of the categorical columns you want to combine to define your groups

3. Run `summarise` saying how you want to combine the quantitative values

4. Run `ungroup` to remove any remaining group information

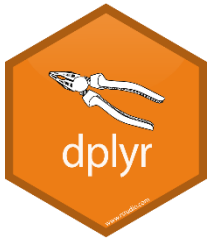
Grouping and Summarising




```
> group.data

# A tibble: 8 x 5
  Sample Genotype Sex   Height Length
   <dbl> <chr>    <chr>  <dbl>  <dbl>
1      1 WT       M         15    200
2      2 WT       F         13    185
3      3 WT       F         14    221
4      4 WT       M         18    265
5      5 KO       M         26    120
6      6 KO       F         22    165
7      7 KO       F         19    143
8      8 KO       M         27    110
```

- Want to get the average Height and Length for each combination of sex and genotype

Grouping and Summarising Workflow



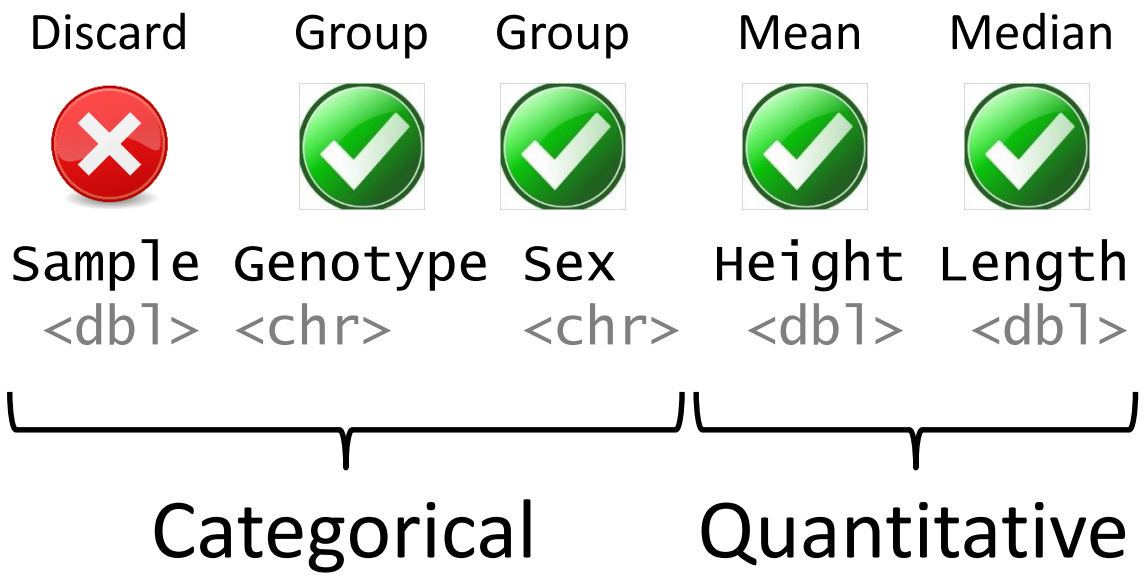
1. Load a tibble with repeated values in one or more columns

2. Use `group_by` to select all of the categorical columns you want to combine to define your groups

3. Run `summarise` saying how you want to combine the quantitative values

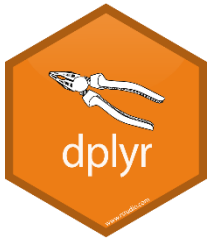
4. Run `ungroup` to remove any remaining group information

Grouping and Summarising



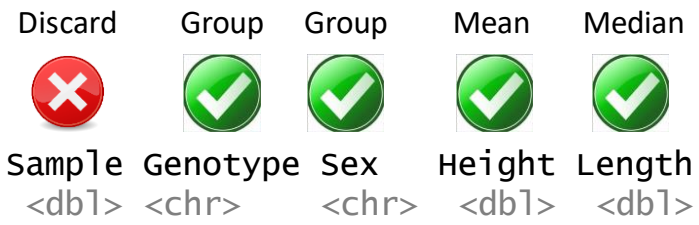
- Want to get the average Height and Length for each combination of sex and genotype

Grouping and Summarising

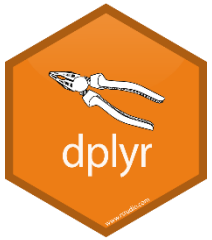



```
group.data %>% group_by(Genotype,Sex)
```

```
# A tibble: 8 x 5
# Groups:   Genotype, Sex [4]
  Sample Genotype Sex   Height Length
   <dbl> <chr>    <chr>  <dbl>  <dbl>
1      1 WT       M         15    200
2      2 WT       F         13    185
3      3 WT       F         14    221
4      4 WT       M         18    265
5      5 KO       M         26    120
6      6 KO       F         22    165
7      7 KO       F         19    143
8      8 KO       M         27    110
```



Grouping and Summarising Workflow



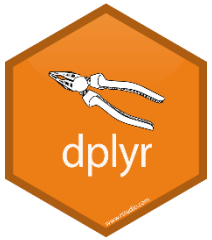
1. Load a tibble with repeated values in one or more columns

2. Use `group_by` to select all of the categorical columns you want to combine to define your groups

3. Run `summarise` saying how you want to combine the quantitative values

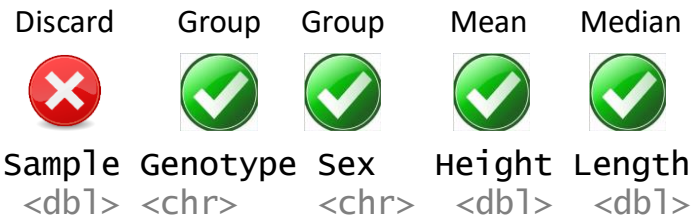
4. Run `ungroup` to remove any remaining group information

Grouping and Summarising

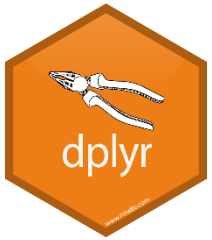



```
group.data %>%
  group_by(Genotype,Sex) %>%
  count()
```

```
# A tibble: 4 x 3
# Groups:   Genotype, Sex [4]
  Genotype Sex       n
  <chr>    <chr> <int>
1 KO       F         2
2 KO       M         2
3 WT       F         2
4 WT       M         2
```



Grouping and Summarising




```
group.data %>%
  group_by(Genotype,Sex) %>%
  summarise(
    Height2=mean(Height),
    Length=median(Length)
  )
```

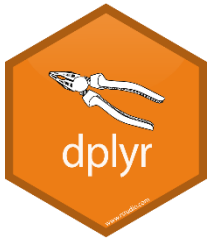
```
# A tibble: 4 x 4
# Groups:   Genotype [2]
  Genotype Sex   Height2 Length
  <chr>    <chr>   <dbl>  <dbl>
1 KO       F        20.5   154
2 KO       M        26.5   115
3 WT       F        13.5   203
4 WT       M        16.5   232.
```

Discard | Group | Group | Mean | Median

Sample	Genotype	Sex	Height	Length
<dbl>	<chr>	<chr>	<dbl>	<dbl>

If you want the count of values as part of a summarised result use the n() function

Grouping and Summarising Workflow



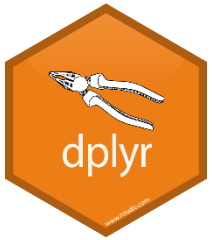
1. Load a tibble with repeated values in one or more columns

2. Use `group_by` to select all of the categorical columns you want to combine to define your groups

3. Run `summarise` saying how you want to combine the quantitative values

4. Run `ungroup` to remove any remaining group information

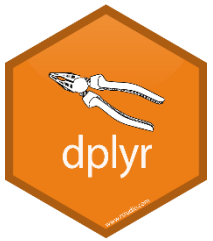
Ungrouping



- A summarise operation removes the the last level of grouping (“Sex” in our worked example)
- Other levels of grouping (“Genotype”) remain annotated on the data, so you could do an additional summarisation if needed
- If you’re not going to use them it’s a good idea to use `ungroup` to remove remaining groups so they don’t interfere with other operations

Grouping affects lots of operations

Find the tallest member of each Sex




```
group.data %>%
  arrange(desc(Height)) %>%
  group_by(Sex) %>%
  slice(1)
```

```
# A tibble: 2 x 5
# Groups:   Sex [2]
  Sample Genotype Sex   Height Length
   <dbl> <chr>    <chr>  <dbl>  <dbl>
1      6 KO       F         22    165
2      8 KO       M         27    110
```

Exercise
Grouping and Summarising

In your restructured tidy2 data find the mean and sem (sem is mean/sqrt(n)) for groups A to E. You'll need to group by sample and then summarise to calculate the two values. You get the number of observations using the n() function

If you have time – use the summarised values to plot a barplot of the data. Use geom_col() and geom_errorbar(). You'll need to set the ymin and ymax aesthetics for geom_errorbar – these will be the mean-sem and mean+sem

Babraham Bioinformatics