tidyverse

Code

tidyverse

David Gresham

Novermber 15, 2017

- Outline
- Load packages
- Tidy data
- Is this table tidy?
- Is this table tidy?
- Is this table tidy?
- Is this table tidy?
- Is this table tidy?
- what are tidy data?
- gather()
- gather()
- spread()
- spread()
- Exercise 1
- Exercise 2
- data import
- Peak at data
- Look at data structure with str()
- Look at the data the tidyverse way
    - using glimpse()
- Assign meaningful names to columns
- Dataframe now has meaningful names
- assign columns proper datatypes
- Select columns using `select()`
- Add a column with `mutate()`
- Sort tibble by column with `arrange()`
- Sort by feature size with arrange()
- Analyze with `summarize()`
- Analyze with `summarize()`
- using the pipe: `%>%`

- using the pipe: `%>%`
    - subset data with `group_by()`
- Filter rows with `filter()`
- Pass dataframe to ggplot for plotting
    - NOTE: ggplot uses `+` not the pipe `%>%`
- Exercise 3
- Exercise 3
- String manipulation with `stringr()`
    - How do we get the gene names?
- Separate values in column with `separate()`
- Combine columns with `unite()`
- Save to a new variable
- Clean up strings with `stringr()`
- Write file
- How do we combine tables?
    - Mutating joins
    - Filtering joins
- Dataset must contain common values (gene names)
- File to join with
- Joining data
- Excercise4
- tidy data don't allow correlation plots
- Rearrange the data
- Plot correlation between t0 and t1
- Excercise5
- example
- Resources

Outline

- tidy data
- reading data
- verbs in dplyr()
    - `select()`
    - `mutate()`
    - `arrange()`
    - `summarise()`
    - `filter()`
- the pipe `%>%`
- joining files

Load packages

Hide

```
library(tidyverse)
```

```
Loading tidyverse: ggplot2
Loading tidyverse: tibble
```

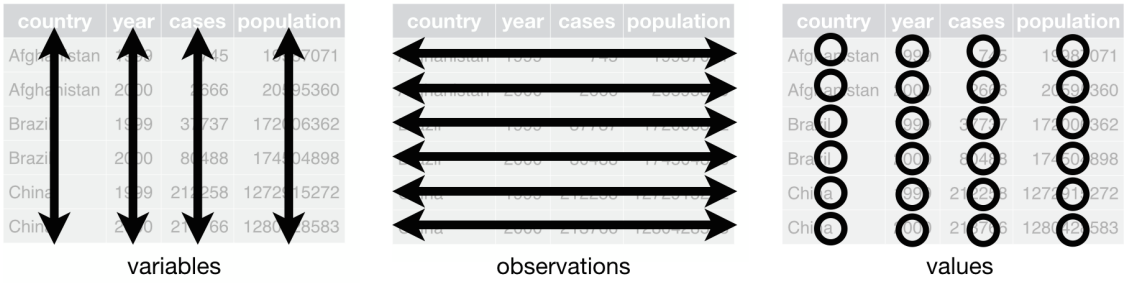
```
Loading tidyverse: tibble
Loading tidyverse: tidyr
Loading tidyverse: readr
Loading tidyverse: purrr
Loading tidyverse: dplyr
Conflicts with tidy packages ---------------------------------------------
---------------------------------------
filter(): dplyr, stats
lag():    dplyr, stats
```

Hide

```
library(stringr)
```

Tidy data

- each **variable** is saved in its on **column**
- each **observation** is saved in its own **row**
- each **value** must have its own **cell**



Is this table tidy?

Hide

```
table2
```

country	year	type	count
<chr>	<int>	<chr>	<int>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898

Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272

1-10 of 12 rows Previous **1** 2 Next

Is this table tidy?

Hide

```
table3
```

	country <chr>	year <int>	rate <chr>
1	Afghanistan	1999	745/19987071
2	Afghanistan	2000	2666/20595360
3	Brazil	1999	37737/172006362
4	Brazil	2000	80488/174504898
5	China	1999	212258/1272915272
6	China	2000	213766/1280428583

6 rows

Is this table tidy?

Hide

```
table1
```

country <chr>	year <int>	cases <int>	population <int>
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

6 rows

Is this table tidy?

Hide

```
gene_expression
```

Gene <chr>	t0 <dbl>	t1 <dbl>	t2 <dbl>
ACT1	2.2	3.2	4.5
GAP1	4.3	2.1	1.6
MEP2	1.6	0.8	0.4
DUR3	-1.2	-1.8	-1.8
MSN2	-2.0	-0.8	-0.1
DAL80	0.2	0.6	0.9

6 rows

Is this table tidy?

Hide

```
facs_data
```

Sample <chr>	Measure <chr>	Value <dbl>
A1	FSC	3.6
A1	FL1	4.5
A2	FSC	3.5
A2	FL1	3.2
A3	FSC	3.8
A3	FL1	4.2

6 rows

what are tidy data?

Jeff Leek in his book **The Elements of Data Analytic Style** summarizes the characteristics of tidy data as the points:

- Each variable you measure should be in one column.
- Each different observation of that variable should be in a different row.
- There should be one table for each “kind” of variable.
- If you have multiple tables, they should include a column in the table that allows them to be linked.

gather()

used when column names are not names of variables, but *values* of a variable (e.g. time).
makes tables *longer* and *skinny* (previously known as melting)

Hide

```
gene_expression
```

Gene <chr>	**t0** <dbl>	**t1** <dbl>	**t2** <dbl>
ACT1	2.2	3.2	4.5
GAP1	4.3	2.1	1.6
MEP2	1.6	0.8	0.4
DUR3	-1.2	-1.8	-1.8
MSN2	-2.0	-0.8	-0.1
DAL80	0.2	0.6	0.9

6 rows

gather()

used when column names are not names of variables, but *values* of a variable (e.g. time).
makes tables *longer* and *skinny* (previously known as melting)

Hide

```
gather(gene_expression, t0:t2, key = "timepoint", value = "expression")
```

Gene <chr>	**timepoint** <chr>	**expression** <dbl>
ACT1	t0	2.2
GAP1	t0	4.3
MEP2	t0	1.6
DUR3	t0	-1.2
MSN2	t0	-2.0
DAL80	t0	0.2
ACT1	t1	3.2
GAP1	t1	2.1
MEP2	t1	0.8

DUR3	t1	-1.8

1-10 of 18 rows Previous **1** 2 Next

spread()

Spreading is the opposite of gathering. Used when an observation is scattered across multiple rows. `spread()` makes tables *shorter* and *wider*

Hide

```
facs_data
```

Sample <chr>	**Measure** <chr>	**Value** <dbl>
A1	FSC	3.6
A1	FL1	4.5
A2	FSC	3.5
A2	FL1	3.2
A3	FSC	3.8
A3	FL1	4.2

6 rows

spread()

Spreading is the opposite of gathering. Used when an observation is scattered across multiple rows. `spread()` makes tables *shorter* and *wider*

Hide

```
spread(facs_data, key = Measure, value = Value)
```

	Sample <chr>	**FL1** <dbl>	**FSC** <dbl>
1	A1	4.5	3.6
2	A2	3.2	3.5
3	A3	4.2	3.8

3 rows

Exercise 1

put table2 in tidy format

Hide

Hide

```
table2[1:6,]
```

country <chr>	year <int>	type <chr>	count <int>
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362

6 rows

	country <chr>	year <int>	cases <int>	population <int>
1	Afghanistan	1999	745	19987071
2	Afghanistan	2000	2666	20595360
3	Brazil	1999	37737	172006362
4	Brazil	2000	80488	174504898
5	China	1999	212258	1272915272
6	China	2000	213766	1280428583

6 rows

Exercise 2

convert table1 to table2

Hide

```
table1
```

country <chr>	year <int>	cases <int>	population <int>
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272

China	2000	213766	1280428583

6 rows

data import

`readr()` has numerous functions for reading in files as tibbles

- 'read_csv()` for comma-delimited
- `read_tsv()` for tab-delimited

Hide

```
gff <- read_delim("Saccharomyces_cerevisiae.R64-1-1.34.gff3",
    "\t", escape_double = FALSE, col_names = FALSE,
    comment = "#", trim_ws = TRUE, skip = 24)
```

```
Parsed with column specification:
cols(
  X1 = col_character(),
  X2 = col_character(),
  X3 = col_character(),
  X4 = col_integer(),
  X5 = col_integer(),
  X6 = col_character(),
  X7 = col_character(),
  X8 = col_character(),
  X9 = col_character()
)
```

Peak at data

a tibble is a dataframe

Hide

```
head(gff)
```

X1 <chr>	X2 <chr>	X3 <chr>	X4 <int>	X5 <int>	X6 <chr>	X7 <chr>	X8 <chr>
I	SGD	CDS	10091	10399	.	+	0
I	SGD	gene	11565	11951	.	-	.
I	SGD	mRNA	11565	11951	.	-	.
I	SGD	exon	11565	11951	.	-	.
I	SGD	CDS	11565	11951	.	-	0
I	SGD	gene	12046	12426	.	+	.

6 rows | 1-8 of 9 columns

Look at data structure with str()

Hide

```
str(gff)
```

```
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':   28848 obs. of  9 variables:
 $ X1: chr  "I" "I" "I" "I" ...
 $ X2: chr  "SGD" "SGD" "SGD" "SGD" ...
 $ X3: chr  "CDS" "gene" "mRNA" "exon" ...
 $ X4: int  10091 11565 11565 11565 11565 12046 12046 12046 12046 13363
...
 $ X5: int  10399 11951 11951 11951 11951 12426 12426 12426 12426 13743
...
 $ X6: chr  "." "." "." "." ...
 $ X7: chr  "+" "-" "-" "-" ...
 $ X8: chr  "0" "." "." "." ...
 $ X9: chr  "ID=CDS:YAL066W;Parent=transcript:YAL066W;protein_id=YAL066W"
"ID=gene:YAL065C;biotype=protein_coding;description=Putative protein of u
nknown function%3B has homology to FLO1%3B possible pse"| __truncated__
"ID=transcript:YAL065C;Parent=gene:YAL065C;biotype=protein_coding;transcr
ipt_id=YAL065C" "Parent=transcript:YAL065C;Name=YAL065C.1;constitutive=1;
ensembl_end_phase=0;ensembl_phase=0;exon_id=YAL065C.1;rank=1" ...
 - attr(*, "spec")=List of 2
  ..$ cols   :List of 9
  .. ..$ X1: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X2: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X3: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X4: list()
  .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
  .. ..$ X5: list()
  .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
  .. ..$ X6: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X7: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X8: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ X9: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  ..$ default: list()
  .. ..- attr(*, "class")= chr  "collector_guess" "collector"
  ..- attr(*, "class")= chr "col_spec"
```

Look at the data the tidyverse way

using glimpse()

Hide

```
glimpse(gff)
```

```
Observations: 28,848
Variables: 9
$ X1 <chr> "I", "I", "I", "I", "I", "I", "I", "I", "I", "I", "I", "I",
"I", "I", "I", "I", "I", "I", "I", "...
$ X2 <chr> "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SGD",
"SGD", "SGD", "SGD", "SGD", "SGD"...
$ X3 <chr> "CDS", "gene", "mRNA", "exon", "CDS", "gene", "mRNA", "exon",
"CDS", "gene", "mRNA", "exon", "CD...
$ X4 <int> 10091, 11565, 11565, 11565, 11565, 12046, 12046, 12046, 12046,
13363, 13363, 13363, 13363, 21566...
$ X5 <int> 10399, 11951, 11951, 11951, 11951, 12426, 12426, 12426, 12426,
13743, 13743, 13743, 13743, 21850...
$ X6 <chr> ".", ".", ".", ".", ".", ".", ".", ".", ".", ".", ".", ".",
".", ".", ".", ".", ".", ".", ".", "...
$ X7 <chr> "+", "-", "-", "-", "-", "+", "+", "+", "+", "-", "-", "-", "-
", "+", "+", "+", "+", "-", "-", "...
$ X8 <chr> "0", ".", ".", ".", "0", ".", ".", ".", "0", ".", ".", ".",
"0", ".", ".", ".", "0", ".", ".", "...
$ X9 <chr> "ID=CDS:YAL066W;Parent=transcript:YAL066W;protein_id=YAL066W",
"ID=gene:YAL065C;biotype=protein_...
```

Assign meaningful names to columns

same approach as naming dataframe columns in base R

Hide

```
names(gff) <- c("chromosome",
                "source",
                "feature",
                "start",
                "stop",
                "unknown1",
                "strand",
                "unknown2",
                "info"
                )
```

Dataframe now has meaningful names

note that tidyverse tries to guess data type

Hide

```
glimpse(gff)
```

```
Observations: 28,848
Variables: 9
$ chromosome <chr> "I", "I", "I", "I", "I", "I", "I", "I", "I", "I", "I",
"I", "I", "I", "I", "I", "I", "I"...
$ source     <chr> "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SG
D", "SGD", "SGD", "SGD", "SGD", "SGD...
$ feature    <chr> "CDS", "gene", "mRNA", "exon", "CDS", "gene", "mRNA",
"exon", "CDS", "gene", "mRNA", "ex...
$ start      <int> 10091, 11565, 11565, 11565, 11565, 12046, 12046, 1204
6, 12046, 13363, 13363, 13363, 1336...
$ stop       <int> 10399, 11951, 11951, 11951, 11951, 12426, 12426, 1242
6, 12426, 13743, 13743, 13743, 1374...
$ unknown1   <chr> ".", ".", ".", ".", ".", ".", ".", ".", ".", ".", ".",
".", ".", ".", ".", ".", ".", "."...
$ strand     <chr> "+", "-", "-", "-", "-", "+", "+", "+", "+", "-", "-",
"-", "-", "+", "+", "+", "+", "-"...
$ unknown2   <chr> "0", ".", ".", ".", "0", ".", ".", ".", "0", ".", ".",
".", "0", ".", ".", ".", "0", "."...
$ info       <chr> "ID=CDS:YAL066W;Parent=transcript:YAL066W;protein_id=Y
AL066W", "ID=gene:YAL065C;biotype=...
```

assign columns proper datatypes

assigning correct data type is critical for anlayses and plotting with ggplot()

Hide

```
gff$feature = as.factor(gff$feature)
gff$chromosome = as.factor(gff$chromosome)
gff$strand = as.factor(gff$strand)
glimpse(gff)
```

```
Observations: 28,848
Variables: 9
$ chromosome <fctr> I, I, I, I, I, I, I, I, I, I, I, I, I, I, I, I, I, I,
I, I, I, I, I, I, I, I, I, I, I, ...
$ source     <chr> "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SGD", "SG
D", "SGD", "SGD", "SGD", "SGD", "SGD...
$ feature    <fctr> CDS, gene, mRNA, exon, CDS, gene, mRNA, exon, CDS, ge
ne, mRNA, exon, CDS, gene, mRNA, e...
$ start      <int> 10091, 11565, 11565, 11565, 11565, 12046, 12046, 1204
6, 12046, 13363, 13363, 13363, 1336...
$ stop       <int> 10399, 11951, 11951, 11951, 11951, 12426, 12426, 1242
6, 12426, 13743, 13743, 13743, 1374...
$ unknown1   <chr> ".", ".", ".", ".", ".", ".", ".", ".", ".", ".", ".",
".", ".", ".", ".", ".", ".", "."...
$ strand     <fctr> +, -, -, -, -, +, +, +, +, -, -, -, -, +, +, +, +, -,
-, -, -, -, -, -, -, +, +, +, +, ...
$ unknown2   <chr> "0", ".", ".", ".", "0", ".", ".", ".", "0", ".", ".",
".", "0", ".", ".", ".", "0", "."...
$ info       <chr> "ID=CDS:YAL066W;Parent=transcript:YAL066W;protein_id=Y
AL066W", "ID=gene:YAL065C;biotype=...
```

Select columns using `select()`

Hide

```
gff <- select(gff, c("chromosome", "feature", "start", "stop", "strand"))
head(gff)
```

chromosome <fctr>	feature <fctr>	start <int>	stop <int>	strand <fctr>
I	CDS	10091	10399	+
I	gene	11565	11951	-
I	mRNA	11565	11951	-
I	exon	11565	11951	-
I	CDS	11565	11951	-
I	gene	12046	12426	+

6 rows

Add a column with `mutate()`

Hide

```
gff <- mutate(gff, length = abs(start - stop))
head(gff)
```

chromosome <fctr>	feature <fctr>	start <int>	stop <int>	strand <fctr>	length <int>
I	CDS	10091	10399	+	308
I	gene	11565	11951	-	386
I	mRNA	11565	11951	-	386
I	exon	11565	11951	-	386
I	CDS	11565	11951	-	386
I	gene	12046	12426	+	380

6 rows

Sort tibble by column with `arrange()`

writing `dplyr::arrange` specifies the package and function

Hide

```
dplyr::arrange(gff,length)
```

chromosome <fctr>	feature <fctr>	start <int>	stop <int>	strand <fctr>	length <int>
IX	exon	155222	155222	+	0
IX	CDS	155222	155222	+	0
X	exon	365784	365784	+	0
X	CDS	365784	365784	+	0
XIII	exon	754220	754220	-	0
XIII	CDS	754220	754220	-	0
XV	exon	901194	901194	-	0
XV	CDS	901194	901194	-	0
II	exon	168423	168424	+	1
II	CDS	168423	168424	+	1

1-10 of 28,848 rows Previous **1** 2 3 4 5 6 ... 100 Next

Sort by feature size with arrange()

sort largest to smallest using -

Hide

```
dplyr::arrange(gff,-length)
```

chromosome <fctr>	feature <fctr>	start <int>	stop <int>	strand <fctr>	length <int>
IV	chromosome	1	1531933	.	1531932
XV	chromosome	1	1091291	.	1091290
VII	chromosome	1	1090940	.	1090939
XII	chromosome	1	1078177	.	1078176
XVI	chromosome	1	948066	.	948065
XIII	chromosome	1	924431	.	924430
II	chromosome	1	813184	.	813183
XIV	chromosome	1	784333	.	784332
X	chromosome	1	745751	.	745750
XI	chromosome	1	666816	.	666815

1-10 of 28,848 rows Previous **1** 2 3 4 5 6 ... 100 Next

Analyze with `summarize()`

this creates a new tibble/dataframe

Hide

```
summarise(gff, mean = mean(length), sd = sd(length), min = min(length), m
ax = max(length), n = n())
```

mean	sd	min	max	n
<dbl>	<dbl>	<dbl>	<dbl>	<int>
1685.739	19480.2	0	1531932	28848

1 row

Analyze with `summarize()`

the function `n()` counts how many observations their are

Hide

```
summarise(gff, mean = mean(length), sd = sd(length), min = min(length), m
ax = max(length), n = n())
```

using the pipe: %>%

- the pipe is from the `magittr` package
- same as `|` in unix
- allows you to perform multiple sequential functions
- pronounced **then**
- unleashes the true power of tidyverse functions

using the pipe: %>%

subset data with `group_by()`

Hide

```
gff %>%
mutate(length = abs(start - stop)) %>%
group_by(feature) %>%
summarise(mean = mean(length), sd = sd(length), min = min(length), max =
max(length), n = n())
```

feature	mean	sd	min	max	n
<fctr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
CDS	1283.28403	1117.68987	0	14732	7045

chromosome	745429.43750	370024.12096	85778	1531932	16
exon	1211.49954	1120.83137	0	14732	7547
gene	1368.93539	1157.35142	50	14732	6686
mRNA	1368.93539	1157.35142	50	14732	6686
ncRNA_gene	761.53333	727.81757	168	3198	15
pseudogene	1254.66667	1503.47151	185	5250	42
rRNA	1837.00000	2141.45001	118	5946	16
rRNA_gene	1837.00000	2141.45001	118	5946	16
snoRNA	181.19481	145.38479	77	1003	77

1-10 of 15 rows Previous **1** 2 Next

Filter rows with `filter()`

Hide

```
gff %>%
filter(feature != "mRNA" & feature != "rRNA_gene" & feature != "snoRNA_ge
ne"& feature != "snRNA_gene") %>%
mutate(length = abs(start - stop)) %>%
group_by(feature) %>%
summarise(mean = mean(length), sd = sd(length), min = min(length), max =
max(length), n = n())
```

feature <fctr>	**mean** <dbl>	**sd** <dbl>	**min** <dbl>	**max** <dbl>	**n** <int>
CDS	1283.28403	1117.68987	0	14732	7045
chromosome	745429.43750	370024.12096	85778	1531932	16
exon	1211.49954	1120.83137	0	14732	7547
gene	1368.93539	1157.35142	50	14732	6686
ncRNA_gene	761.53333	727.81757	168	3198	15
pseudogene	1254.66667	1503.47151	185	5250	42
rRNA	1837.00000	2141.45001	118	5946	16
snoRNA	181.19481	145.38479	77	1003	77
snRNA	400.33333	412.99379	111	1174	6
transcript	111.15287	212.43146	70	3198	314

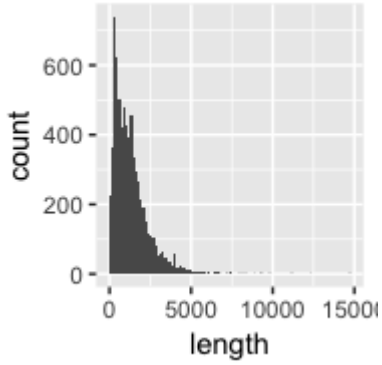
1-10 of 11 rows Previous **1** 2 Next

Pass dataframe to ggplot for plotting

NOTE: ggplot uses + not the pipe %>%

Hide

```
gff %>%
filter(feature == c("CDS")) %>%
ggplot(aes(x = length)) +
        geom_histogram(bins = 100)
```



Exercise 3

plot the population of each country in 1999 using `%>%` and ggplot()`

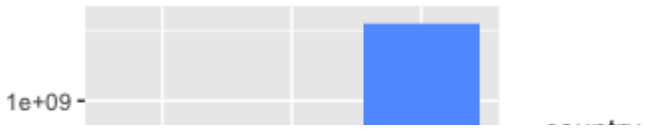
Hide

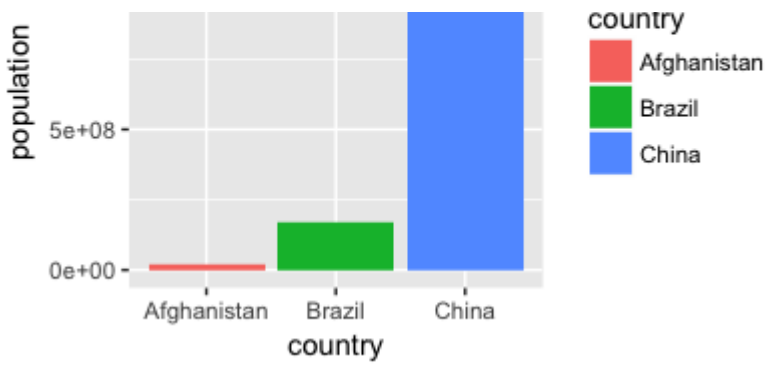
```
table1
```

country <chr>	year <int>	cases <int>	population <int>
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

6 rows

Exercise 3





String manipulation with `stringr()`

How do we get the gene names?

Hide

```
select(gff, info)
```

info <chr>
ID=CDS:YAL066W;Parent=transcript:YAL066W;protein_id=YAL066W
ID=gene:YAL065C;biotype=protein_coding;description=Putative protein of unknown function% homology to FLO1%3B possible pseudogene [Source:SGD%3BAcc:S000001817];gene_id=YAL065C;logic_name=sgd
ID=transcript:YAL065C;Parent=gene:YAL065C;biotype=protein_coding;transcript_id=YAL065C
Parent=transcript:YAL065C;Name=YAL065C.1;constitutive=1;ensembl_end_phase=0;ensembl
ID=CDS:YAL065C;Parent=transcript:YAL065C;protein_id=YAL065C
ID=gene:YAL064W-B;biotype=protein_coding;description=Fungal-specific protein of unknown [Source:SGD%3BAcc:S000002141];gene_id=YAL064W-B;logic_name=sgd
ID=transcript:YAL064W-B;Parent=gene:YAL064W-B;biotype=protein_coding;transcript_id=YAL
Parent=transcript:YAL064W-B;Name=YAL064W- B.1;constitutive=1;ensembl_end_phase=0;ensembl_phase=0;exon_id=YAL064W-B.1;rank=1
ID=CDS:YAL064W-B;Parent=transcript:YAL064W-B;protein_id=YAL064W-B
ID=gene:YAL064C-A;Name=TDA8;biotype=protein_coding;description=Putative protein of unk function%3B null mutant is sensitive to expression of the top1-T722A allele%3B not an essent [Source:SGD%3BAcc:S000002140];gene_id=YAL064C-A;logic_name=sgd

1-10 of 28,848 rows Previous **1** 2 3 4 5 6 … 100 Next

Separate values in column with `separate()`

Hide

```
gff %>%
mutate(length = abs(start - stop)) %>%
filter(feature == "gene") %>%
separate(col = "info", into = c("info1", "info2", "info3", "info4", "info
5"), sep = ";", extra = "merge") %>%
separate(col = "info1", into = c("junk", "Systematic_name"), sep = ":") %
>%
separate(col = "info2", into = c("junk2", "Gene"), sep = "Name=") %>%
separate(col = "info3", into = c("junk3", "Description1"), sep = "descrip
tion=") %>%
separate(col = "info4", into = c("junk4", "Description2"), sep = "descrip
tion=") %>%
select(c(Description1, Description2))
```

1
2
3
4
5
6
7
8
9
10

1-10 of 6,686 rows | 1-1 of 2 columns Previous **1** 2 3 4 5 6 … 100 Next

Combine columns with `unite()`

Hide

```
gff %>%
mutate(length = abs(start - stop)) %>%
filter(feature == "gene") %>%
separate(col = "info", into = c("info1", "info2", "info3", "info4", "info
5"), sep = ";", extra = "merge") %>%
separate(col = "info1", into = c("junk", "Systematic_name"), sep = ":") %
>%
separate(col = "info2", into = c("junk2", "Gene"), sep = "Name=") %>%
separate(col = "info3", into = c("junk3", "Description1"), sep = "descrip
tion=") %>%
separate(col = "info4", into = c("junk4", "Description2"), sep = "descrip
tion=") %>%
```

```
unite(Description, Description1, Description2, sep = "") %>%
select(c( Description))
```

1
2
3
4
5
6
7
8
9
10

1-10 of 6,686 rows | 1-1 of 1 columns Previous **1** 2 3 4 5 6 … 100 Next

Save to a new variable

A general rule is if you are piping more than 10 steps save as a new variable

Hide

```
gff_clean <- gff %>%
mutate(length = abs(start - stop)) %>%
filter(feature == "gene") %>%
separate(col = "info", into = c("info1", "info2", "info3", "info4", "info
5"), sep = ";", extra = "merge") %>%
separate(col = "info1", into = c("junk", "Systematic_name"), sep = ":") %
>%
separate(col = "info2", into = c("junk2", "Gene"), sep = "Name=") %>%
separate(col = "info3", into = c("junk3", "Description1"), sep = "descrip
tion=") %>%
separate(col = "info4", into = c("junk4", "Description2"), sep = "descrip
tion=") %>%
unite(Description, Description1, Description2, sep = "") %>%
select(c(Systematic_name, Gene, Description))
```

Clean up strings with `stringr()`

Hide

```
gff_clean$Description <- str_replace_all(gff_clean$Description, "%3B", ""
)
```

```
gff_clean$Description <- str_replace_all(gff_clean$Description, "%2C", ""
)
gff_clean$Description <- str_replace_all(gff_clean$Description, "^NA", ""
)

gff_clean %>%
select(c(Description))
```

Write file

Hide

```
write_tsv(gff_clean, "Yeast_genes.txt", na = "NA")
```

How do we combine tables?

Mutating joins

A mutating join allows you combine variables from two tables by matchiung observations by their keys

1. Inner Join

matches pairs of observation from two tables whenever their keys are equal

2. Outer join

keeps observations that appear in at least one of the tables

- left join keeps all the observations in x (should be the default)
- right join keeps all the observations in y
- full join keeps all observations in x and y

Filtering joins

affects (filters) the observations not the variables

- semi_join(x, y) keeps all observations in x that have a match in y
- anti_join(x, y) drops all observations in x that have a match in y

Dataset must contain common values (gene names)

Hide

```
str(data)
```

File to join with

Hide

```
str(gff_clean)
```

Joining data

`dplyr::left_join(a, b, by = "x1")` Join matching rows from b to a.

Hide

```
left_join(gff_clean, data, by = c("Systematic_name" = "Syst")) %>%
        str()
```

Excercise4

tidy data don’t allow correlation plots

Hide

```
tidy_gene_expression <- gene_expression %>%
        gather(t0:t2, key = "timepoint", value = "expression")
tidy_gene_expression
```

Gene <chr>	**timepoint** <chr>	**expression** <dbl>
ACT1	t0	2.2
GAP1	t0	4.3
MEP2	t0	1.6
DUR3	t0	-1.2
MSN2	t0	-2.0
DAL80	t0	0.2
ACT1	t1	3.2
GAP1	t1	2.1
MEP2	t1	0.8
DUR3	t1	-1.8

1-10 of 18 rows Previous **1** 2 Next

Rearrange the data

Plot correlation between t0 and t1

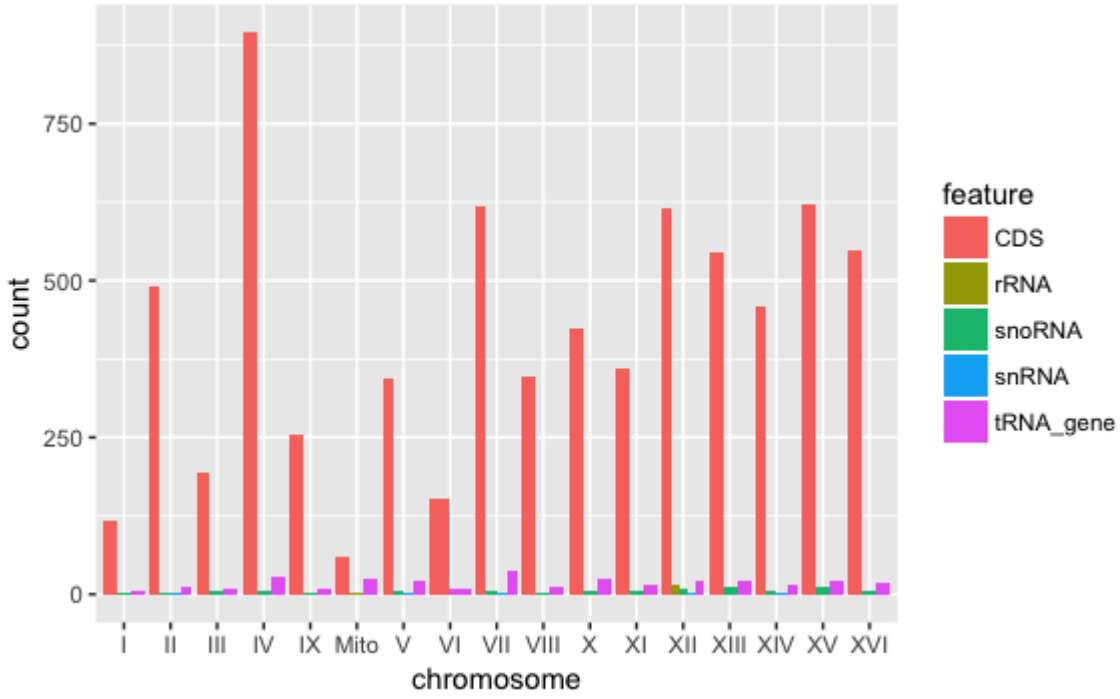
Excercise5

- Get the gff file for your favorite organism from the SGR
- Tidy the gff
- plot distribution of features per chromosome

example

Hide

```
ggplot(data = yeast_features, mapping = aes(x = chromosome, fill = featur
e)) +
        geom_bar(position = "dodge")
```



April 11, 2018 mkhalfan