Wide to Long Data with facets in ggplot2

Youmi Suk

School of Data Science, University of Virginia

2. 14. 2022

Example data

- Our example data is taken from a much larger dataset collected by the Minneapolis Public School District (MPLS) in minnesota (USA).
- The variables in the data include reading achievement scores from grades 5 to 8, risk group (risk), gender (gen), ethnicity (eth), English language learner status (ell), special education services (sped), and attendance proportion (att).

```
dat <- read.table("https://studysites.sagepub.com/long/chapters/datas
kable(head(dat))
```

subid	read.5	read.6	read.7	read.8	risk	gen	eth	ell	sped	att
1	172	185	179	194	HHM	F	Afr	0	N	0.94
2	200	210	209	-99	HHM	F	Afr	0	N	0.91
3	191	199	203	215	HHM	M	Afr	0	N	0.97
4	200	195	194	-99	HHM	F	Afr	0	N	0.88
5	207	213	212	213	HHM	F	Afr	0	N	0.85
6	191	189	206	195	HHM	M	Afr	0	N	0.90

Relationship between reading scores and attendance proportions

- How can we create scatterplots between each reading score (read.5 to read.8) and attendance proportions (att)?
- First, draw a scatterplot between read.5 to att.

```
ggplot(dat, aes(x=att, y=read.5)) + geom_point() + theme_bw()
```

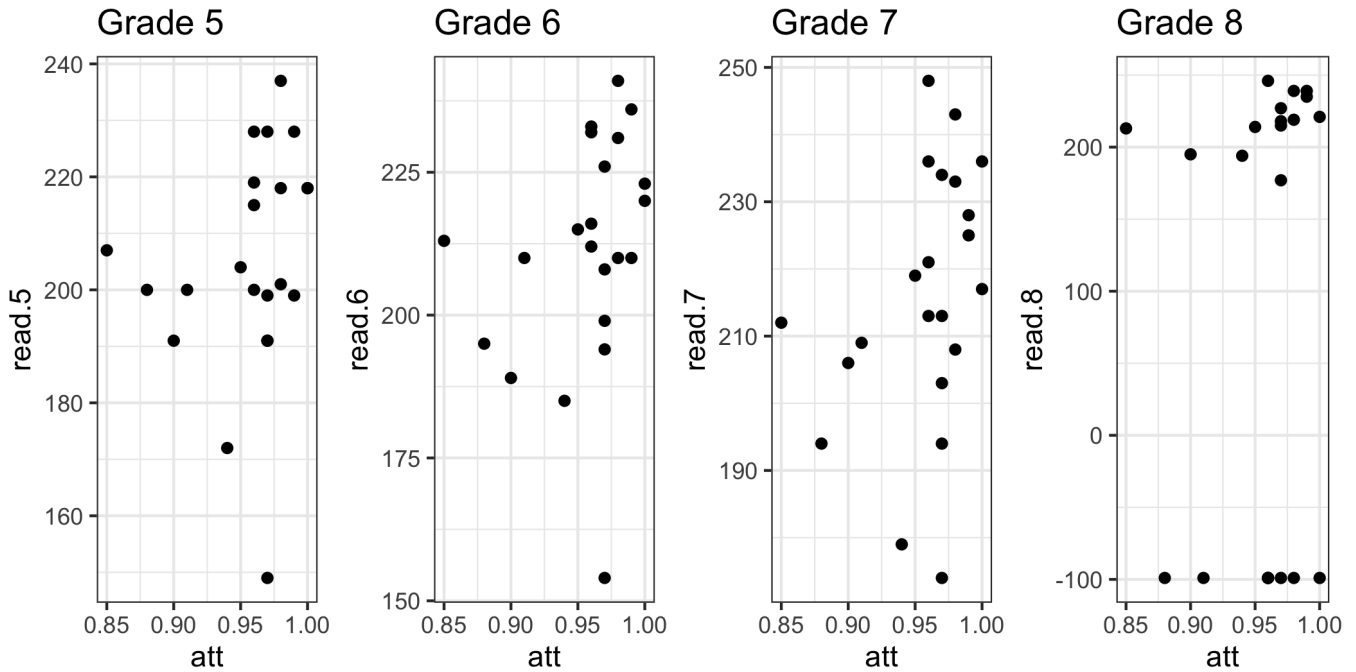
Relationship between reading scores and attendance proportions

- We may want to draw separate scatterplots for different reading measures and combine separate plots by using `gridExtra::grid.arrange()`, `ggpubr::ggarrange()`, `cowplot::plot_grid()`, or `patchwork`.

```
p1 <- ggplot(dat, aes(x=att, y=read.5)) + geom_point() +
   labs(title="Grade 5") + theme_bw()
p2 <- ggplot(dat, aes(x=att, y=read.6)) + geom_point() +
   labs(title="Grade 6") + theme_bw()
p3 <- ggplot(dat, aes(x=att, y=read.7)) + geom_point() +
   labs(title="Grade 7") + theme_bw()
p4 <- ggplot(dat, aes(x=att, y=read.8)) + geom_point() +
   labs(title="Grade 8") + theme_bw()

grid.arrange(p1, p2, p3, p4, nrow=1) # from package gridExtra
```

Relationship between reading scores and attendance proportions

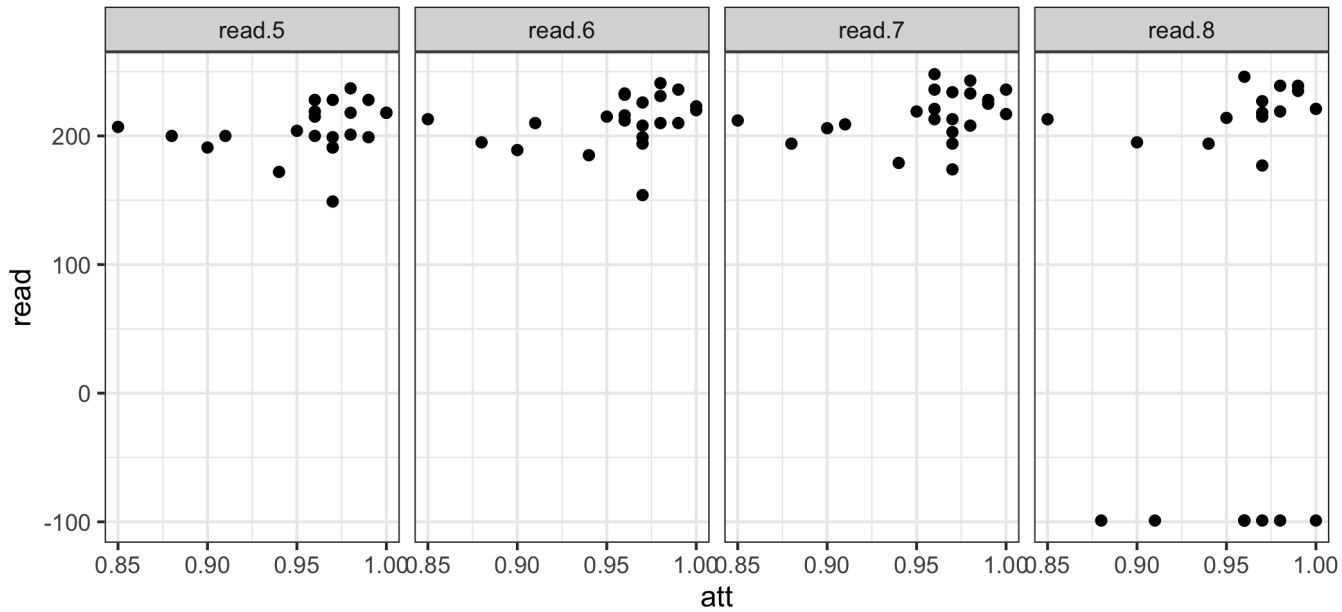
- We may want to draw separate scatterplots for different reading measures and combine separate plots by using `gridExtra::grid.arrange()`, `ggpubr::ggarrange()`, `cowplot::plot_grid()`, or `patchwork`.



Can we do this by using only one *ggplot* function?

- Of course, but we have to change our data format.

```
dat2 <- gather(dat, key = "grade", value= "read", read.5:read.8) # pa
ggplot(dat2, aes(x=att, y=read)) + geom_point() +
   facet_wrap( ~ grade, nrow=1) + theme_bw()
```



Wide vs. Long format

- When processing and plotting data, how you choose your columns can have a great impact on how easy your data is to manipulate. Data can either be in ‘long’ (or ‘tidy’) form, or it can be in wide form. Some plotting libraries are designed to work with ‘long’ data, and others with wide data.
- A simple difference between the wide-form and the long-form is that the wide-form displays many measurements from one individual in one row and the column names show what the measurements are.

Wide format

- Wide format is the standard structure, and it is sometimes referred to as a *subjects-by-variables* format or *multivariate* format. For longitudinal data, wide format has the characteristics that data collected at different time points appear in multiple columns. In our example data, the reading scores appear in four columns (read.5 to read.8), reflecting the repeated measureds aspect of the reading variable. A static variable, such as attendance (att), occupies only a single column because it is measured at a single occasion.

```
head(dat) # wide format
```

```
##   subid read.5 read.6 read.7 read.8 risk gen eth ell sped  att
## 1     1    172    185    179    194  HHM   F Afr   0    N 0.94
## 2     2    200    210    209    -99  HHM   F Afr   0    N 0.91
## 3     3    191    199    203    215  HHM   M Afr   0    N 0.97
## 4     4    200    195    194    -99  HHM   F Afr   0    N 0.88
## 5     5    207    213    212    213  HHM   F Afr   0    N 0.85
## 6     6    191    189    206    195  HHM   M Afr   0    N 0.90
```

Long format

- The main feature of long format is that the repeated measures of the subjects appear vertically and are stacked one atop another. Long format is sometimes referred to as univariate format because the response variable occupies a single column. Static variables appear in additional columns with their values repeated for the duration of time because their values do not change over time.

```
head(dat2) # long format
```

```
##   subid risk gen eth ell sped  att  grade read
## 1     1  HHM   F Afr   0    N 0.94 read.5  172
## 2     2  HHM   F Afr   0    N 0.91 read.5  200
## 3     3  HHM   M Afr   0    N 0.97 read.5  191
## 4     4  HHM   F Afr   0    N 0.88 read.5  200
## 5     5  HHM   F Afr   0    N 0.85 read.5  207
## 6     6  HHM   M Afr   0    N 0.90 read.5  191
```

Another Example: *iris* data from base R

The `iris` data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

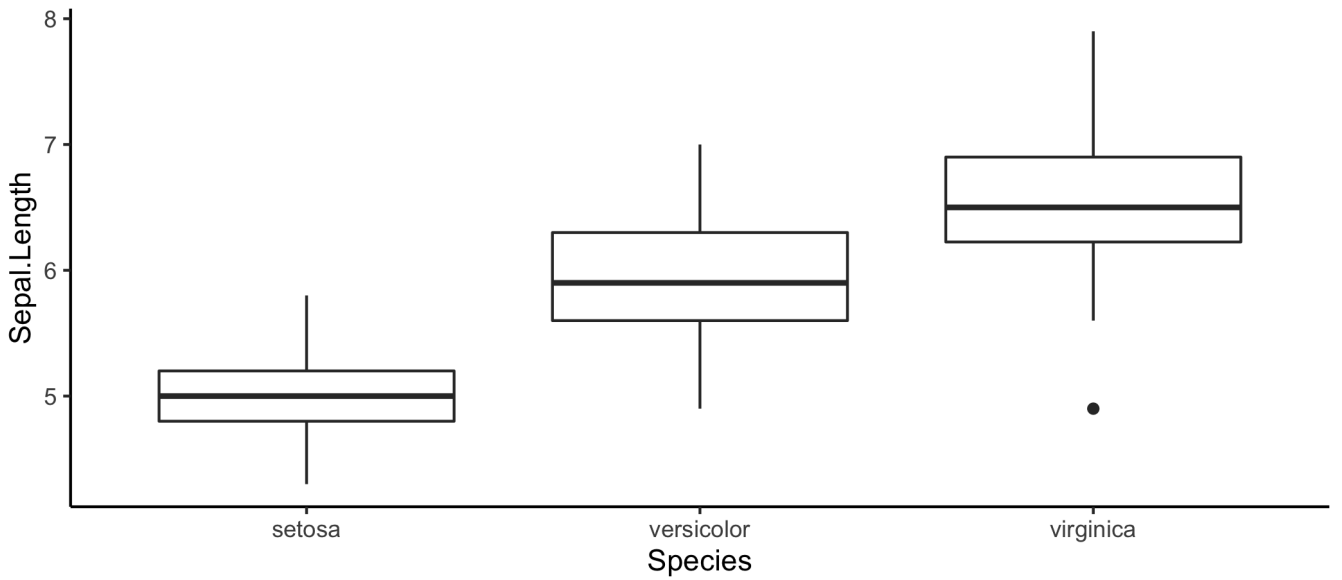
```
head(iris, 10)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1           5.1         3.5          1.4         0.2  setosa
## 2           4.9         3.0          1.4         0.2  setosa
## 3           4.7         3.2          1.3         0.2  setosa
## 4           4.6         3.1          1.5         0.2  setosa
## 5           5.0         3.6          1.4         0.2  setosa
## 6           5.4         3.9          1.7         0.4  setosa
## 7           4.6         3.4          1.4         0.3  setosa
## 8           5.0         3.4          1.5         0.2  setosa
## 9           4.4         2.9          1.4         0.2  setosa
## 10          4.9         3.1          1.5         0.1  setosa
```

Q1: Relationships between species and the four measures

- Check how sepal length and width and petal length and width are related to species.
- What plots can we use?

```
ggplot(iris, aes(x=Species, y=Sepal.Length)) + geom_boxplot() + theme
```



Q1: Relationships between species and the four measures

- Gather the four variables: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width. There are three ways to gather the variables.

```
# Way1
longiris1 <- gather(iris, key = "flower_att", value = "measurement",
                    Sepal.Length, Sepal.Width, Petal.Length, Petal.W
# Way 2
longiris2 <- gather(iris, key = "flower_att", value = "measurement",
                    Sepal.Length:Petal.Width)
# Way 3
longiris3 <- gather(iris, key = "flower_att", value = "measurement",
                    -Species)
```

Q1: Relationships between species and the four measures

- Gather the four variables: Sepal.Length, Sepal.Width, Petal.Length, Petal.Width. There are three ways to gather the variables.

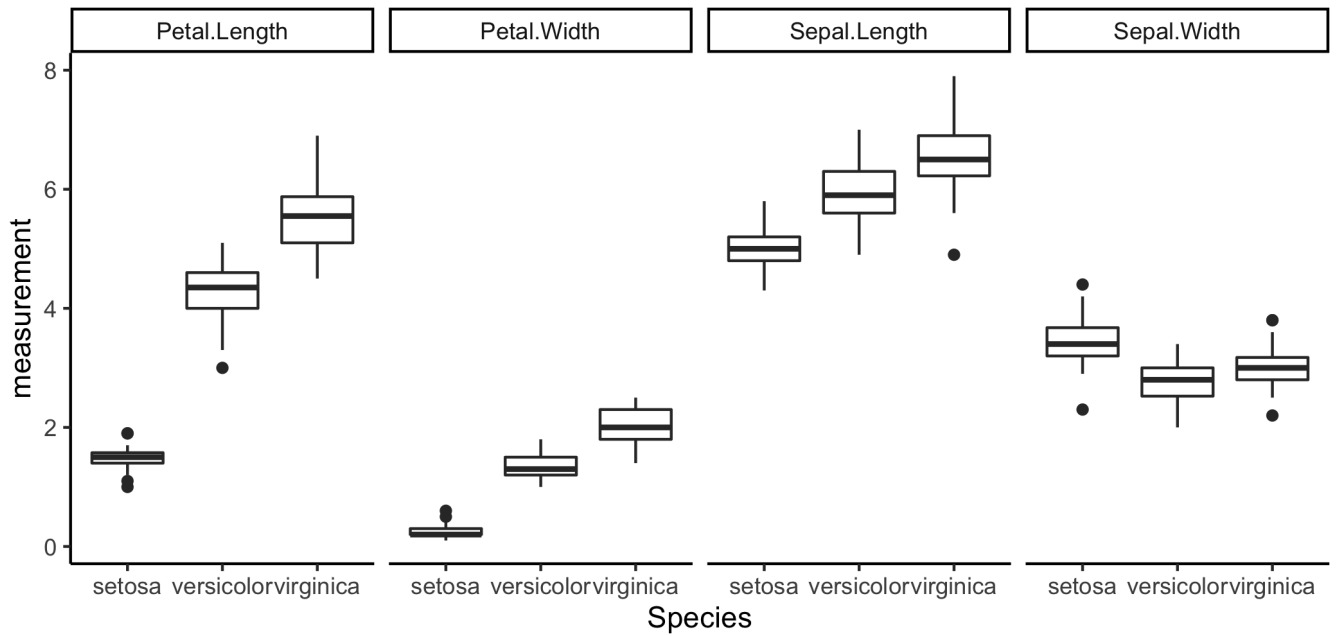
```
head(longiris1, 10)
```

```
##    Species   flower_att measurement
## 1   setosa Sepal.Length         5.1
## 2   setosa Sepal.Length         4.9
## 3   setosa Sepal.Length         4.7
## 4   setosa Sepal.Length         4.6
## 5   setosa Sepal.Length         5.0
## 6   setosa Sepal.Length         5.4
## 7   setosa Sepal.Length         4.6
## 8   setosa Sepal.Length         5.0
## 9   setosa Sepal.Length         4.4
## 10  setosa Sepal.Length         4.9
```

Q1: Relationships between species and the four measures

- Use fuction `facet_...` in `ggplot2`.

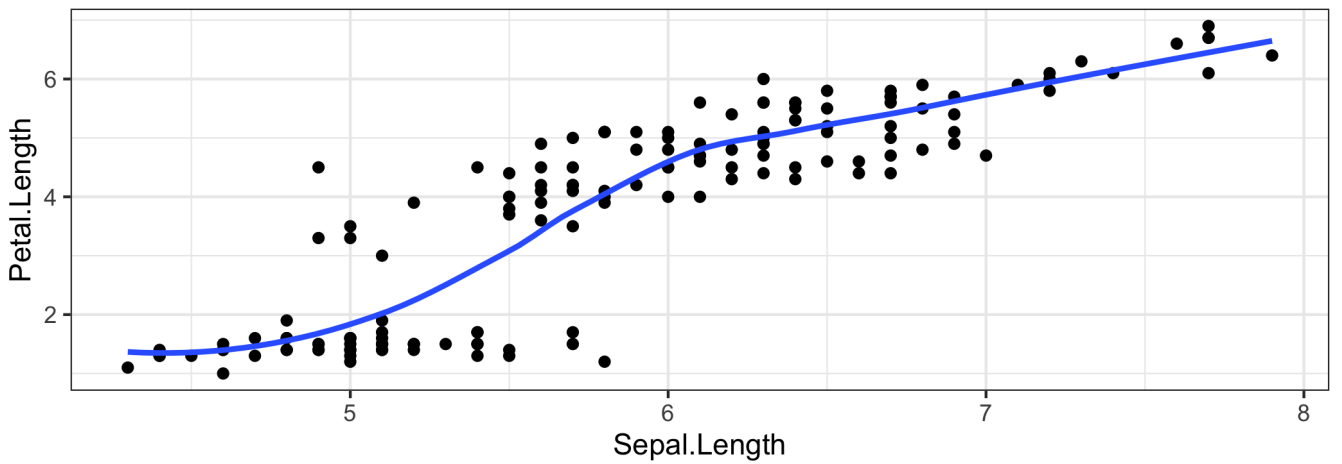
```
ggplot(longiris1, aes(x=Species, y=measurement)) + geom_boxplot() +
   facet_grid(~ flower_att) + theme_classic()
```



Q2: Relationships between sepal variables and petal variables

- Check how sepal variables (length and width) are related to pedal variables (length and width).
- One example:

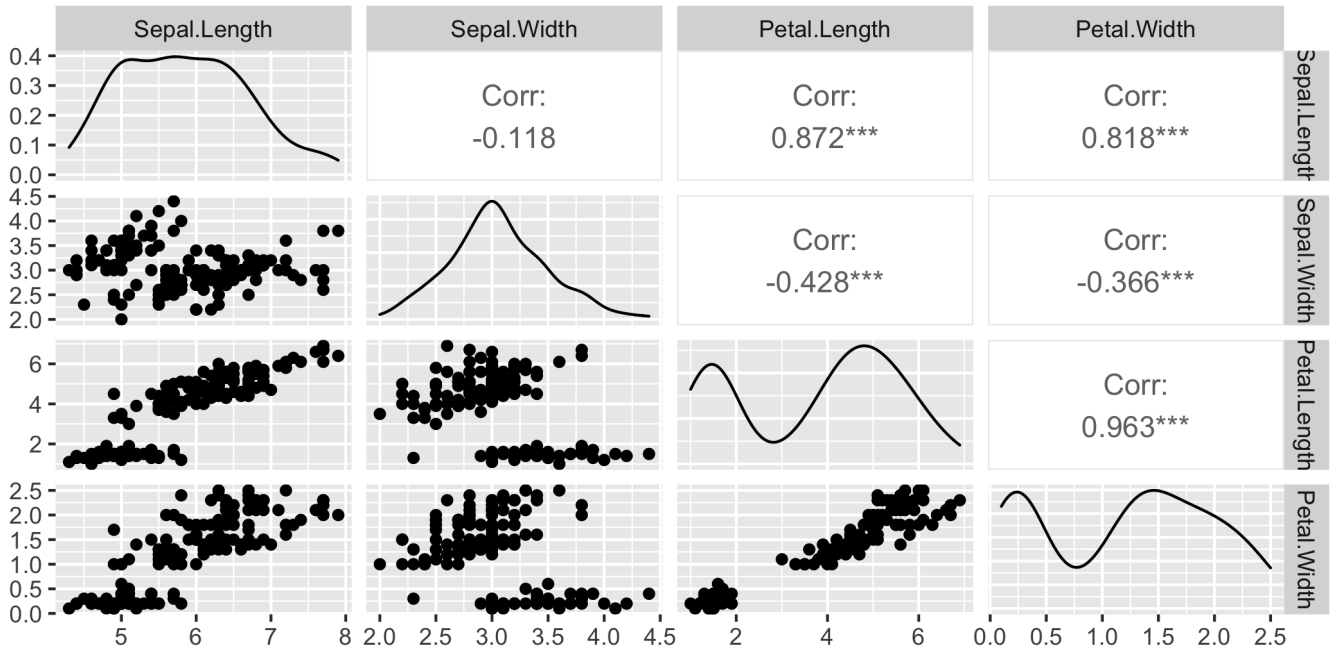
```
ggplot(iris, aes(x=Sepal.Length, y=Petal.Length)) + geom_point() +
   geom_smooth(method="loess", formula = y ~ x, se =F) + theme_bw()
```



Q2: Relationships between sepal variables and petal variables

- We might want to use a scatterplot matrix.

```
library(GGally)
ggpairs(iris[, c("Sepal.Length", "Sepal.Width", "Petal.Length", "Peta
```



Q2: Relationships between sepal variables and petal variables

- Gather Sepal variables first and then gather Petal variables.

```
liris <- gather(iris, key = "Sepal", value = "Sepal.measures",
                Sepal.Length, Sepal.Width)
liris2 <- gather(liris, key = "Petal", value = "Petal.measures",
                 Petal.Length, Petal.Width)

head(liris2, 8)
```

```
##   Species        Sepal Sepal.measures        Petal Petal.measures
## 1  setosa Sepal.Length             5.1 Petal.Length            1.4
## 2  setosa Sepal.Length             4.9 Petal.Length            1.4
## 3  setosa Sepal.Length             4.7 Petal.Length            1.3
## 4  setosa Sepal.Length             4.6 Petal.Length            1.5
## 5  setosa Sepal.Length             5.0 Petal.Length            1.4
## 6  setosa Sepal.Length             5.4 Petal.Length            1.7
## 7  setosa Sepal.Length             4.6 Petal.Length            1.4
## 8  setosa Sepal.Length             5.0 Petal.Length            1.5
```

Q2: Relationships between sepal variables and petal variables

- Gather Sepal variables first and then gather Petal variables.

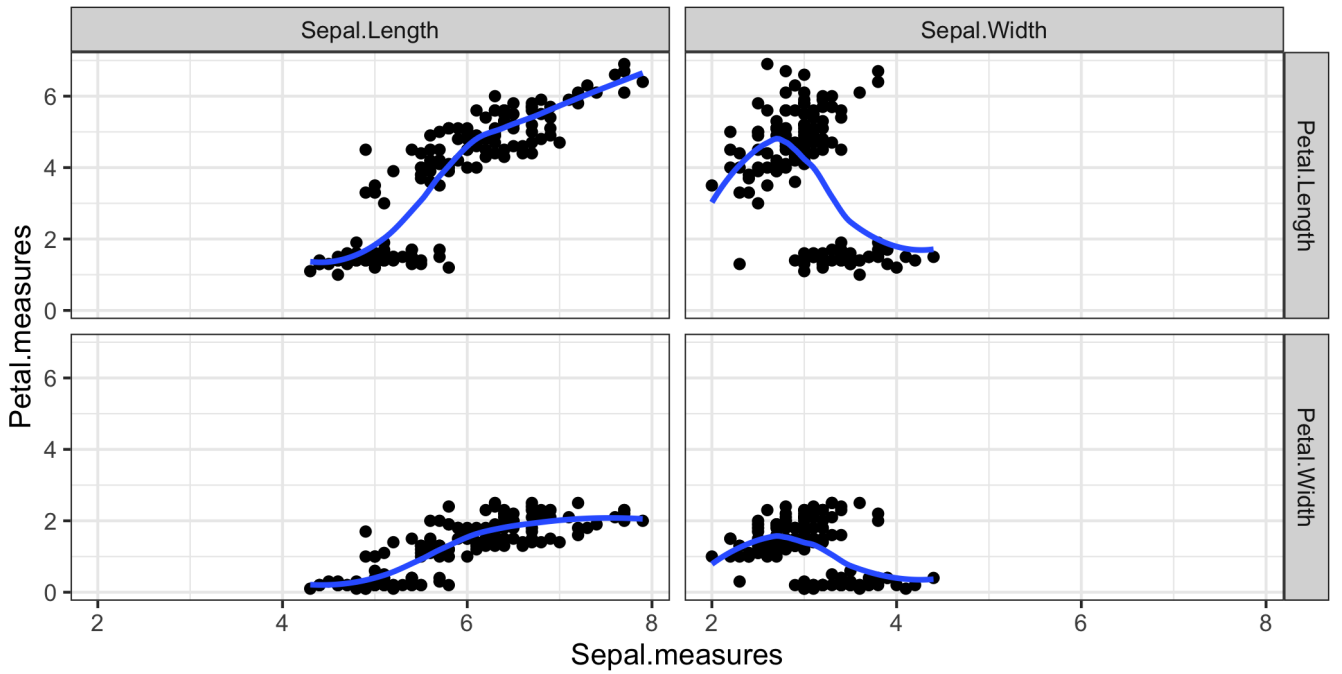
```
# you could use the pipe operator to produce the same result
liris3 <- iris %>%
          gather(key = "Sepal", value = "Sepal.measures", Sepal.Lengt
          gather(key = "Petal", value = "Petal.measures",  Petal.Leng

head(liris3, 8)
```

```
##   Species        Sepal Sepal.measures        Petal Petal.measures
## 1  setosa Sepal.Length            5.1 Petal.Length            1.4
## 2  setosa Sepal.Length            4.9 Petal.Length            1.4
## 3  setosa Sepal.Length            4.7 Petal.Length            1.3
## 4  setosa Sepal.Length            4.6 Petal.Length            1.5
## 5  setosa Sepal.Length            5.0 Petal.Length            1.4
## 6  setosa Sepal.Length            5.4 Petal.Length            1.7
## 7  setosa Sepal.Length            4.6 Petal.Length            1.4
## 8  setosa Sepal.Length            5.0 Petal.Length            1.5
```

Q2: Relationships between sepal variables and petal variables

- Use function `ggplot` with a long-format dataset.

```
ggplot(liris2, aes(x=Sepal.measures, y=Petal.measures)) + geom_point(
   facet_grid(Petal ~ Sepal) + geom_smooth(method="loess", formula =
```



Useful R functions from Wide to Long (or vice versa)

func	package	to_long_form	to_wide_form
stack/unstack	utils	stack	unstack
reshape	stats	reshape(direction='long', ...)	reshape(direction='wide', ...)
melt/dcast	reshape2	melt	dcast
gather/spread	tidyr	gather	spread

Example: Long to Wide format

- Let's revisit a subset of MPLS data.

```
longdat <- gather(dat, key = "grade", value= "read", read.5:read.8) #
head(longdat)
```

```
##   subid risk gen eth ell sped  att  grade read
## 1     1  HHM   F Afr   0    N 0.94 read.5  172
## 2     2  HHM   F Afr   0    N 0.91 read.5  200
## 3     3  HHM   M Afr   0    N 0.97 read.5  191
## 4     4  HHM   F Afr   0    N 0.88 read.5  200
## 5     5  HHM   F Afr   0    N 0.85 read.5  207
## 6     6  HHM   M Afr   0    N 0.90 read.5  191
```

Example: Long to Wide format

- Transform MPLS data from a long to a wide format

```
widedat <- spread(longdat, grade, read) # long to wide format
head(widedat)
```

```
##   subid risk gen eth ell sped  att read.5 read.6 read.7 read.8
## 1     1  HHM   F Afr   0    N 0.94    172    185    179    194
## 2     2  HHM   F Afr   0    N 0.91    200    210    209    -99
## 3     3  HHM   M Afr   0    N 0.97    191    199    203    215
## 4     4  HHM   F Afr   0    N 0.88    200    195    194    -99
## 5     5  HHM   F Afr   0    N 0.85    207    213    212    213
## 6     6  HHM   M Afr   0    N 0.90    191    189    206    195
```