Testing for normality in R

Avalon C.S. Owens, Eric R. Scott

10/26/2018

Packages for today

```
library(tidyverse)
library(car)
skylight = read.csv("skylight.csv")
titanic = read.csv("titanic.csv")
```

Plan for today

- Transforming your data
  - `mutate()` to add columns
  - `filter()` to select rows
- Testing your test assumptions
  - `shapiro.test()` for normality
  - `leveneTest()` for equal variance
- Normal probability plots with `ggplot2`
- Non-parametric tests (part 1 of 2)
  - `wilcox.test()`

Transforming your data

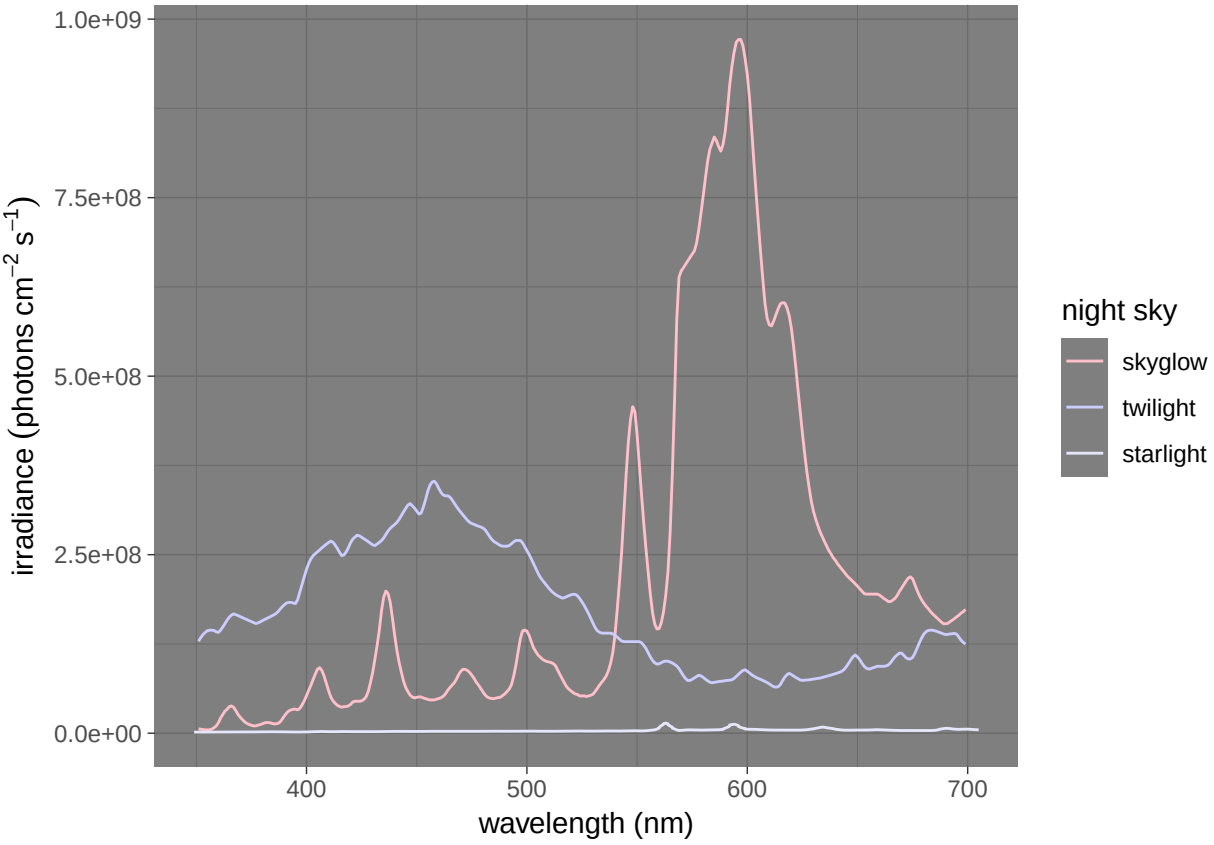
Transforming your data

What to do when your data don't look like you'd like...?

```
skylight = read.csv("skylight.csv")
skylight$sky_type <- factor(skylight$sky_type,
                            levels = c("skyglow",
                                       "twilight",
                                       "starlight"))
str(skylight)
```

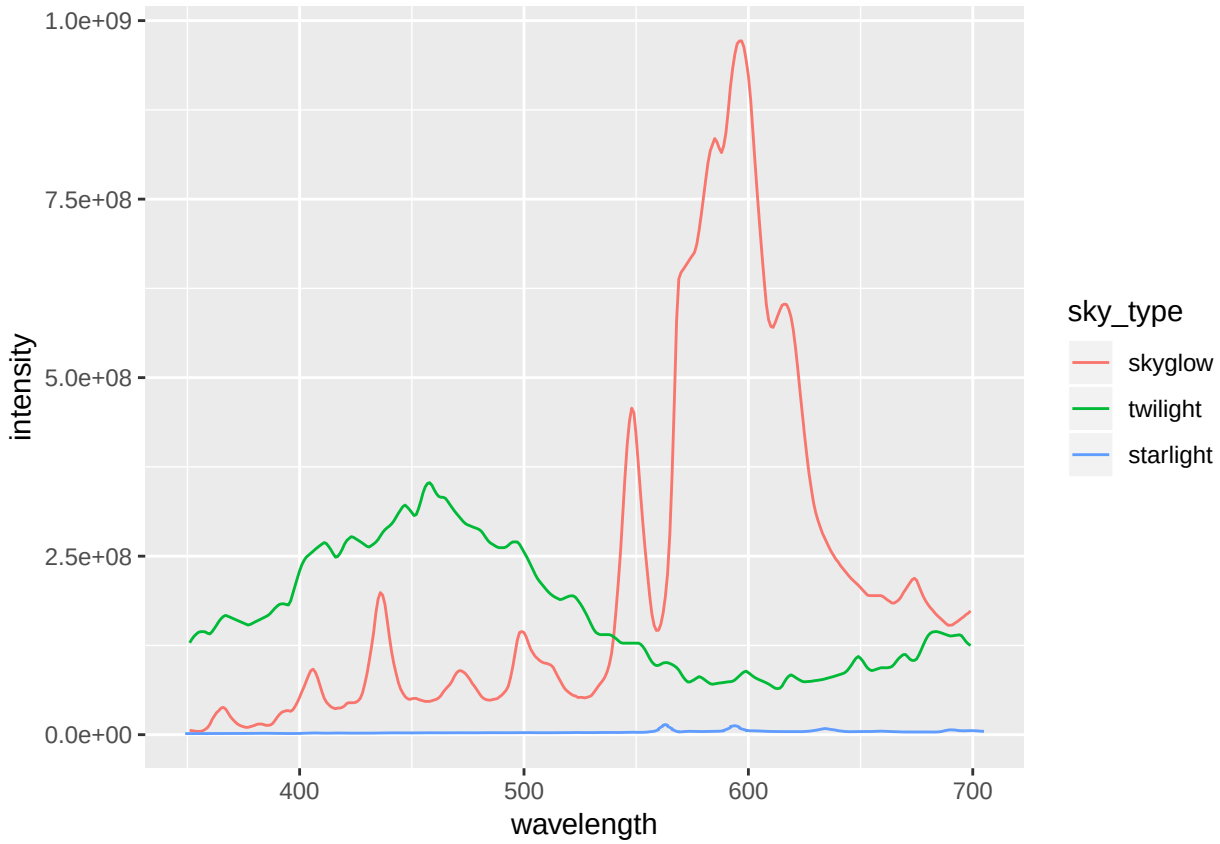
```
## 'data.frame':    841 obs. of  3 variables:
##  $ sky_type  : Factor w/ 3 levels "skyglow","twilight",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ wavelength: int  349 354 362 369 376 380 384 388 392 394 ...
##  $ intensity : num  1517171 1539927 1586467 1658928 1709065 ...
```

Transforming your data



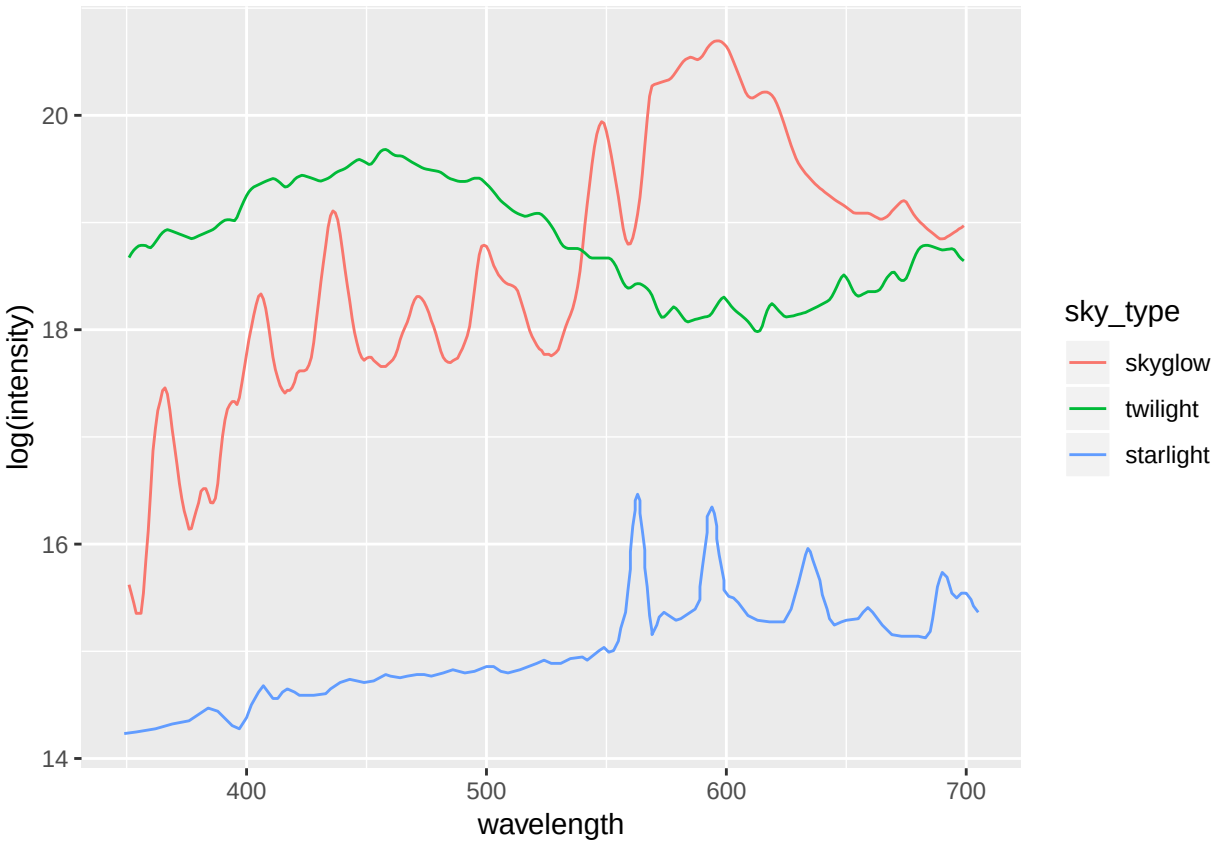
Transforming your data

```
ggplot(skylight, aes(x = wavelength, y = intensity,
                     color = sky_type)) + geom_line()
```



Transforming your data

```
ggplot(skylight, aes(x = wavelength, y = log(intensity),
                     color = sky_type)) + geom_line()
```



Transforming your data with `mutate()`

```
head(skylight)
```

```
##    sky_type wavelength intensity
## 1 starlight        349   1517171
## 2 starlight        354   1539927
## 3 starlight        362   1586467
## 4 starlight        369   1658928
## 5 starlight        376   1709065
## 6 starlight        380   1813931
```

```
log_skylight <- skylight %>% mutate(log_intensity = log(intensity))
```

Transforming your data with `mutate()`

```
head(log_skylight)
```

```
##    sky_type wavelength intensity log_intensity
## 1 starlight        349   1517171      14.23236
## 2 starlight        354   1539927      14.24725
## 3 starlight        362   1586467      14.27702
## 4 starlight        369   1658928      14.32168
## 5 starlight        376   1709065      14.35146
## 6 starlight        380   1813931      14.41101
```

```
log_skylight <- skylight %>% mutate(log_intensity = log(intensity))
```

Transforming your data

Create new column that gives you intensity in **Watts**!

Watts = (speed of light / wavelength in m) x photon count x Planck's constant

```
c <- 3e+08 #speed of light
h <- 6.626e-34 #Planck's constant
nm_to_m <- 1e-09 #nm to m conversion
head(skylight)
```

```
##    sky_type wavelength intensity
## 1 starlight        349   1517171
## 2 starlight        354   1539927
## 3 starlight        362   1586467
## 4 starlight        369   1658928
## 5 starlight        376   1709065
## 6 starlight        380   1813931
```

Transforming your data

```
c <- 3e+08 #speed of light
h <- 6.626e-34 #Planck's constant
nm_to_m <- 1e-09 #nm to m conversion
head(skylight, 2)
```

```
##    sky_type wavelength intensity
## 1 starlight        349   1517171
## 2 starlight        354   1539927
```

```
skylight_W <- skylight %>%
  mutate(intensity_W = (c/wavelength*nm_to_m)*(intensity)*h)
head(skylight_W, 2)
```

```
##    sky_type wavelength intensity  intensity_W
## 1 starlight        349   1517171 8.641353e-31
## 2 starlight        354   1539927 8.647079e-31
```

Coding hack!

Logical operators

$>$: `>`

$<$: `<`

$\geq$: `>=`

$\leq$: `<=` Not the same as `<-`!

$=$: `==`

≠: `!=`

and: `&`

or: `|` Doesn't mean 'given that'!

Logical operators

```
10 <= 6
```

```
## [1] FALSE
```

Logical operators

```
10 <= 6
```

```
## [1] FALSE
```

```
9 == 9
```

```
## [1] TRUE
```

Logical operators

```
10 <= 6
```

```
## [1] FALSE
```

```
9 == 9
```

```
## [1] TRUE
```

```
9 != 7000000
```

```
## [1] TRUE
```

Logical operators

```
10 <= 6
```

```
## [1] FALSE
```

```
9 == 9
```

```
## [1] TRUE
```

```
9 != 7000000
```

```
## [1] TRUE
```

```
9 = 8.999999
```

Logical operators

```
10 <= 6
```

```
## [1] FALSE
```

```
9 == 9
```

```
## [1] TRUE
```

```
9 != 7000000
```

```
## [1] TRUE
```

```
9 = 8.999999
```

```
## Error in 9 = 8.999999: invalid (do_set) left-hand side to assignment
```

Logical operators

```
10 < 11 & 10 > 9
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

```
## [1] TRUE
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

```
## [1] TRUE
```

```
10 < 11 & 10 < 9
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

```
## [1] TRUE
```

```
10 < 11 & 10 < 9
```

```
## [1] FALSE
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

```
## [1] TRUE
```

```
10 < 11 & 10 < 9
```

```
## [1] FALSE
```

```
4 < 2 | 42 > 4
```

Logical operators

```
10 < 11 & 10 > 9
```

```
## [1] TRUE
```

```
1 < 2 | 1 > 3
```

```
## [1] TRUE
```

```
10 < 11 & 10 < 9
```

```
## [1] FALSE
```

```
4 < 2 | 42 > 4
```

```
## [1] TRUE
```

Transforming your data with `filter()`

```
str(skylight)
```

```
## 'data.frame':    841 obs. of  3 variables:
##  $ sky_type  : Factor w/ 3 levels "skyglow","twilight",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ wavelength: int  349 354 362 369 376 380 384 388 392 394 ...
##  $ intensity : num  1517171 1539927 1586467 1658928 1709065 ...
```

```
sky_twilight <- skylight %>% filter(sky_type == "twilight")
```

Transforming your data with `filter()`

```
head(sky_twilight,3)
```

```
##   sky_type wavelength intensity
## 1 twilight        351 128700537
## 2 twilight        352 133797252
## 3 twilight        353 137970647
```

```
sky_twilight <- skylight %>% filter(sky_type == "twilight")
```

Transforming your data with `filter()`

```
head(sky_twilight,3)
```

```
##   sky_type wavelength intensity
## 1 twilight        351 128700537
## 2 twilight        352 133797252
## 3 twilight        353 137970647
```

```
sky_twilight <- skylight %>% filter(sky_type == "twilight")
```

Transforming your data with `filter()`

1. Filter the skylight dataset for 'natural' measurements – anything that wasn't taken during exposure to urban skyglow.
2. BONUS CHALLENGE! Filter the dataset for skyglow and twilight measurements at wavelengths below 500 nm.

(And here's a fun fact: these wavelengths cannot be detected by the satellites used to photograph the earth at night!)

Transforming your data with `filter()`

1. Filter the skylight dataset for 'natural' measurements – anything that wasn't taken during exposure to urban skyglow.

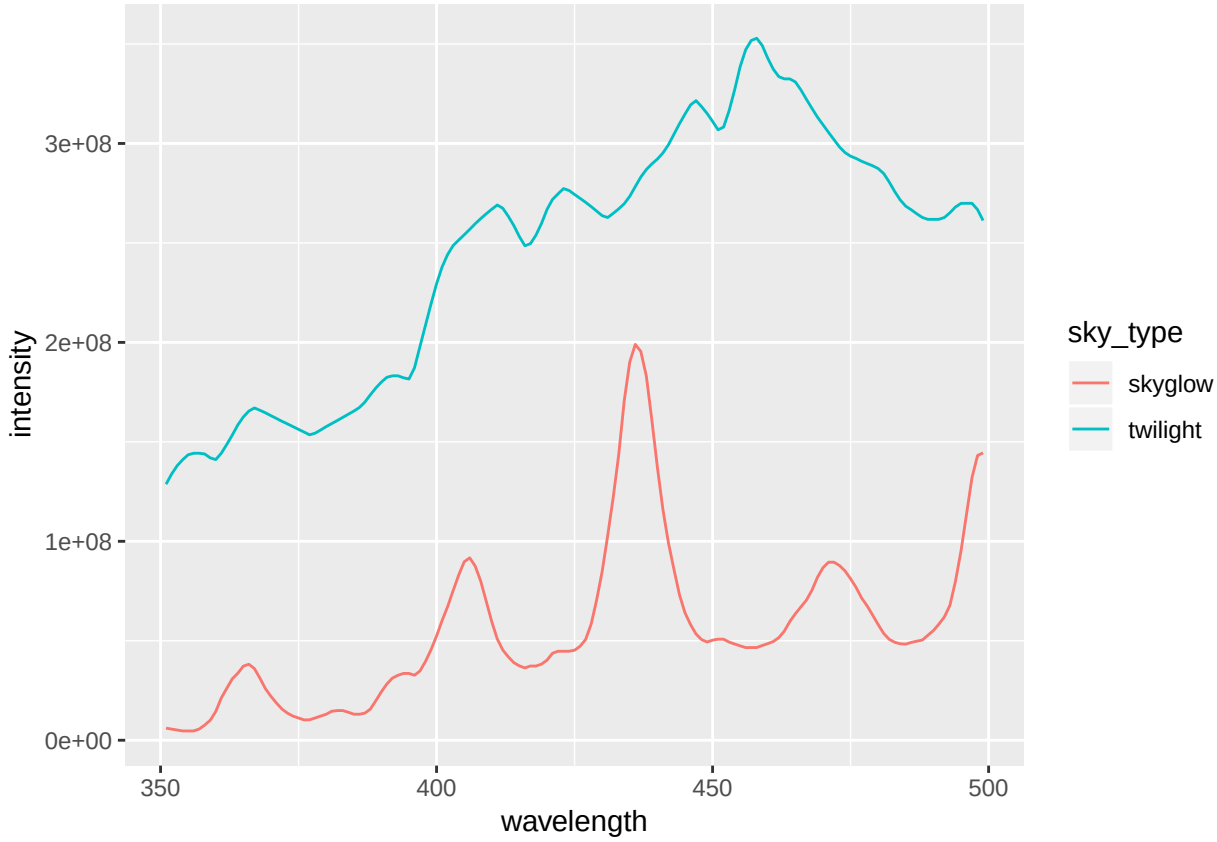
```
sky_natural <- skylight %>%
  filter(sky_type == "twilight" | sky_type == "starlight")
```

2. BONUS CHALLENGE! Filter the dataset for skyglow and twilight measurements at wavelengths below 500 nm.

```
sky_satellite <- skylight %>%
  filter(wavelength < 500 &
         (sky_type == "twilight" | sky_type == "skyglow"))
```

Transforming your data with `filter()`

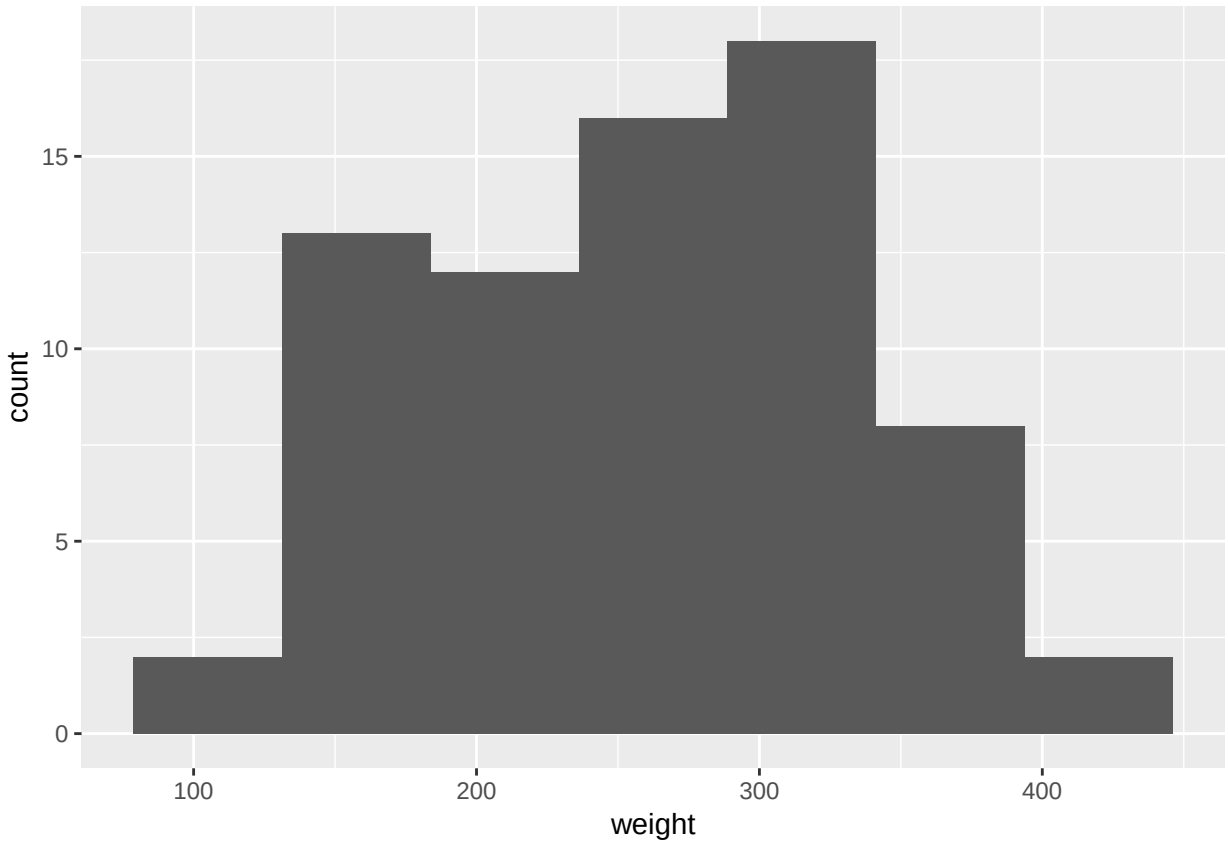
```
ggplot(sky_satellite, aes(x = wavelength, y = intensity,
                          color = sky_type)) + geom_line()
```



Testing your test assumptions

Testing for normality

```
ggplot(chickwts, aes(x = weight)) + geom_histogram(bins = 7)
```



Testing for normality

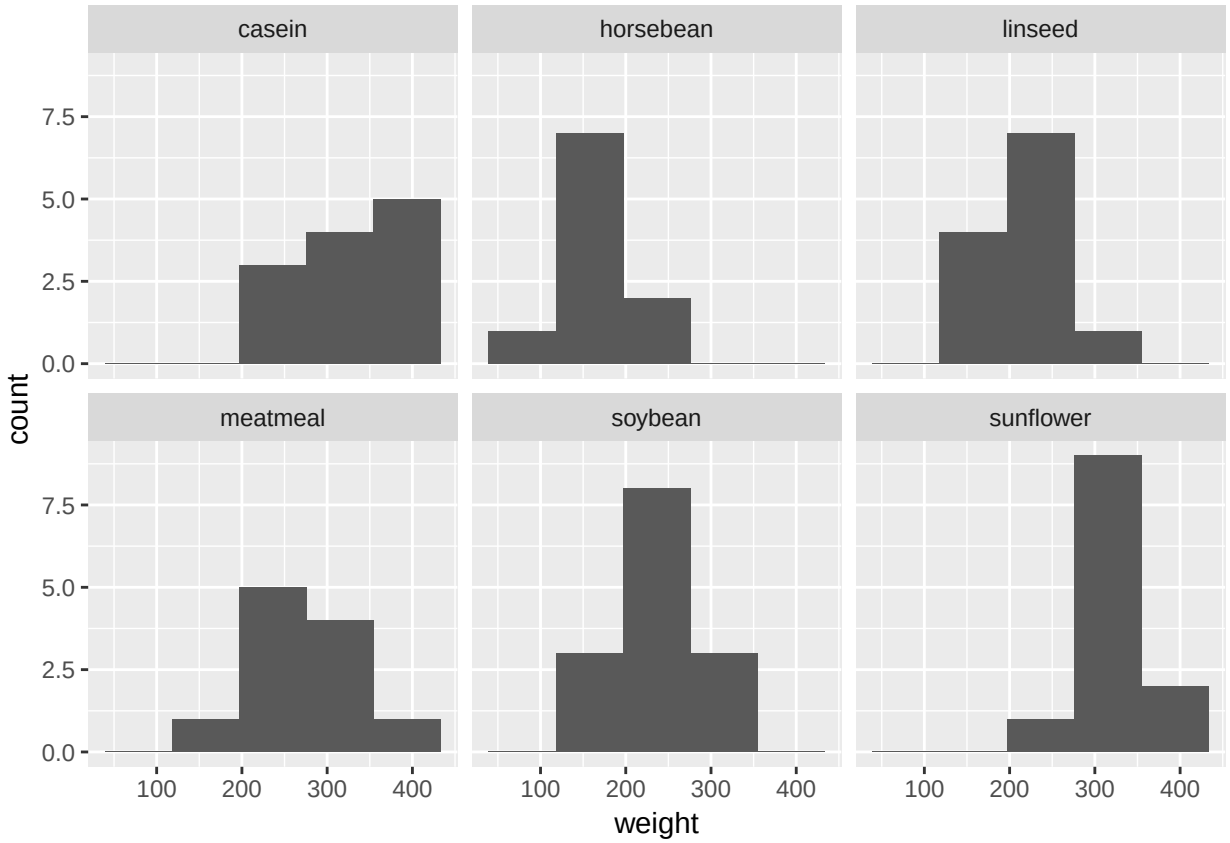
```
shapiro.test(chickwts$weight) #you can do this

chick_weight <- chickwts$weight
shapiro.test(chick_weight) #or this, if you're fancy
```

Testing for normality

```
shapiro.test(chickwts$weight)
```

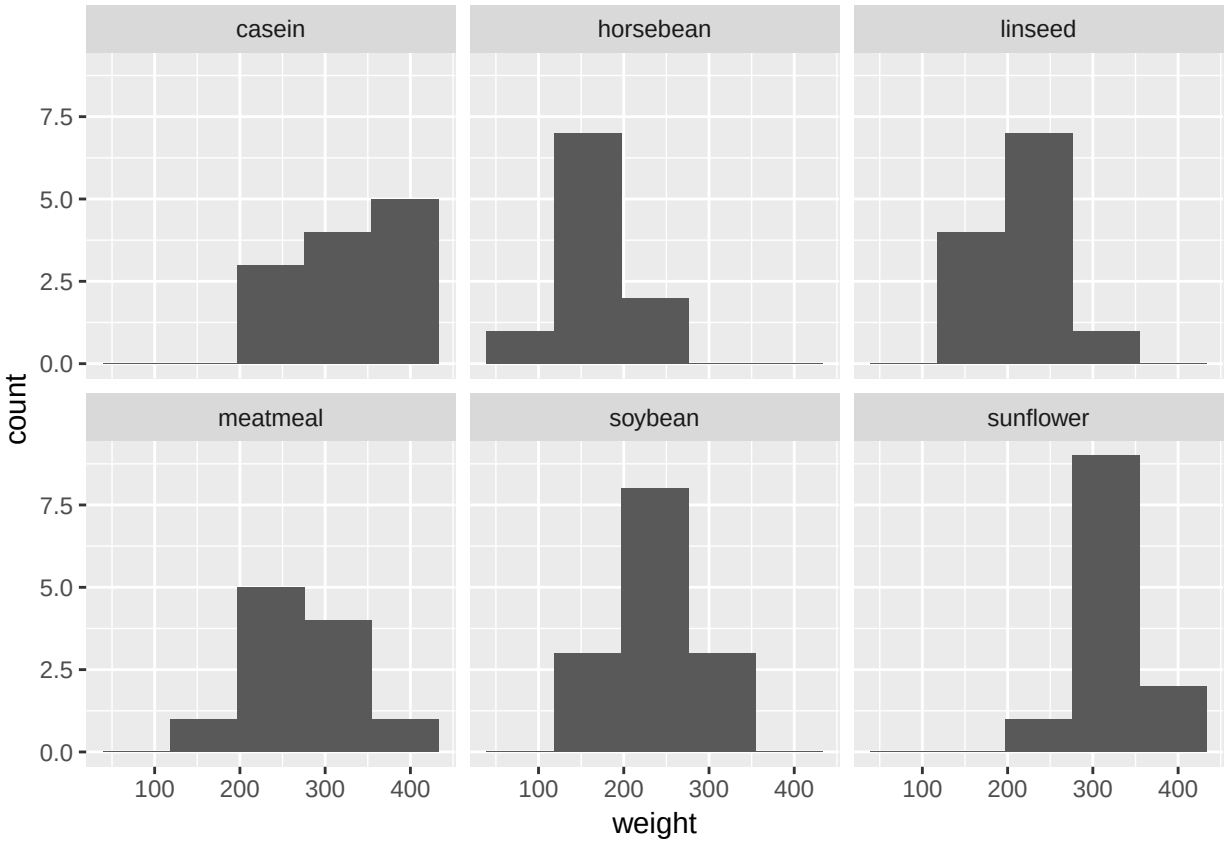
```
##
##  Shapiro-Wilk normality test
##
## data:  chickwts$weight
## W = 0.97674, p-value = 0.2101
```

Testing for normality




```
ggplot(chickwts, aes(x = weight)) +
  geom_histogram(bins = 5) + facet_wrap("feed")
```

Testing for normality



Test animal (meatmeal, casein) and vegetable (everything else) feed types for normality separately!

Testing for normality

```
chick_animal <- chickwts %>%
  filter(feed == "meatmeal" | feed == "casein") %>%
  mutate(diet = "animal")
chick_veggie <- chickwts %>%
  filter(feed != "meatmeal" & feed != "casein") %>%
  mutate(diet = "veggie")
shapiro.test(chick_animal$weight)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  chick_animal$weight
## W = 0.9641, p-value = 0.5508
```

```
shapiro.test(chick_veggie$weight)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  chick_veggie$weight
```

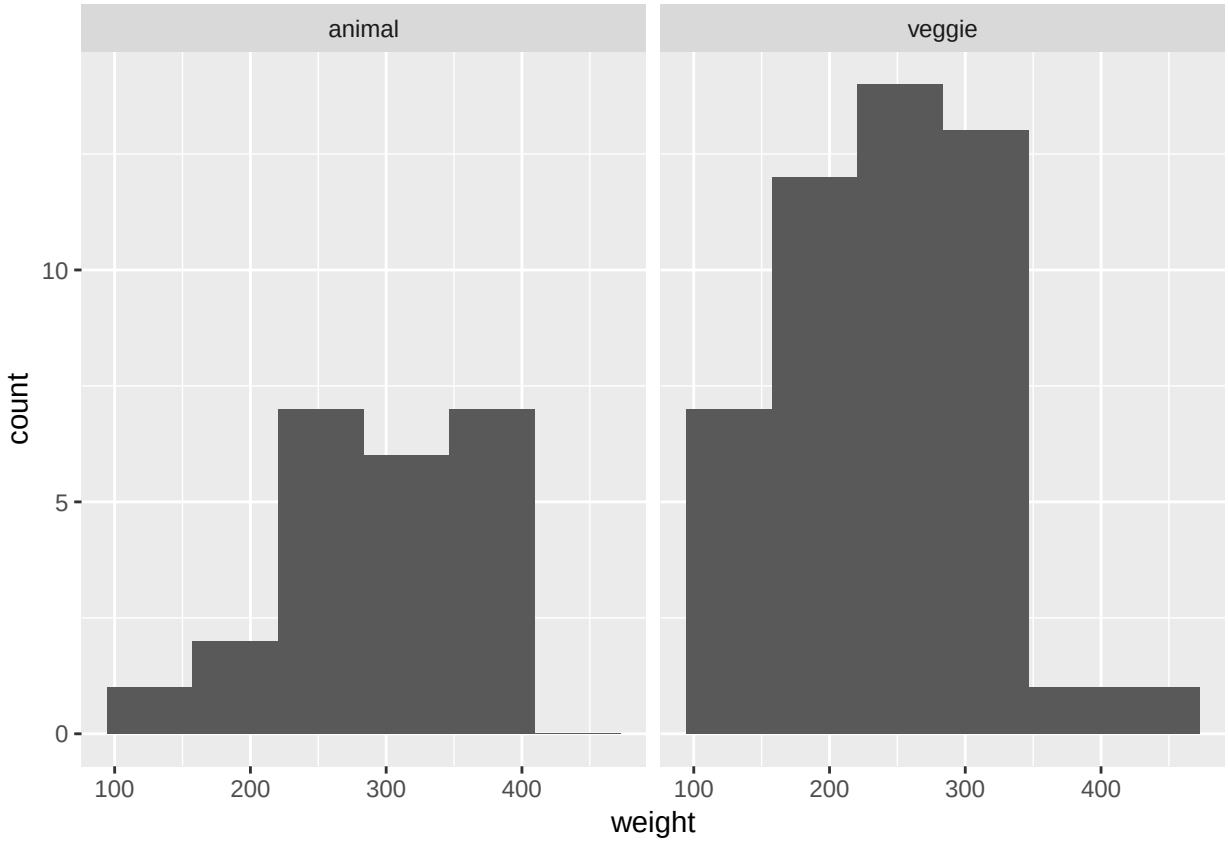
```
## W = 0.97078, p-value = 0.2715
```

Testing for normality

```
chick_animal <- chickwts %>%
  filter(feed == "meatmeal" | feed == "casein") %>%
  mutate(diet = "animal")
chick_veggie <- chickwts %>%
  filter(feed != "meatmeal" & feed != "casein") %>%
  mutate(diet = "veggie")
chick_diet = bind_rows(chick_animal, chick_veggie)
head(chick_diet)
```

```
##   weight     feed   diet
## 1    325 meatmeal animal
## 2    257 meatmeal animal
## 3    303 meatmeal animal
## 4    315 meatmeal animal
## 5    380 meatmeal animal
## 6    153 meatmeal animal
```

Testing for normality



Testing for equal variance

```
library(car)
leveneTest(weight~diet, data = chick_diet)
```

Testing for for equal variance

```
library(car)
leveneTest(weight~diet, data = chick_diet)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  1  0.4226 0.5178
##       69
```

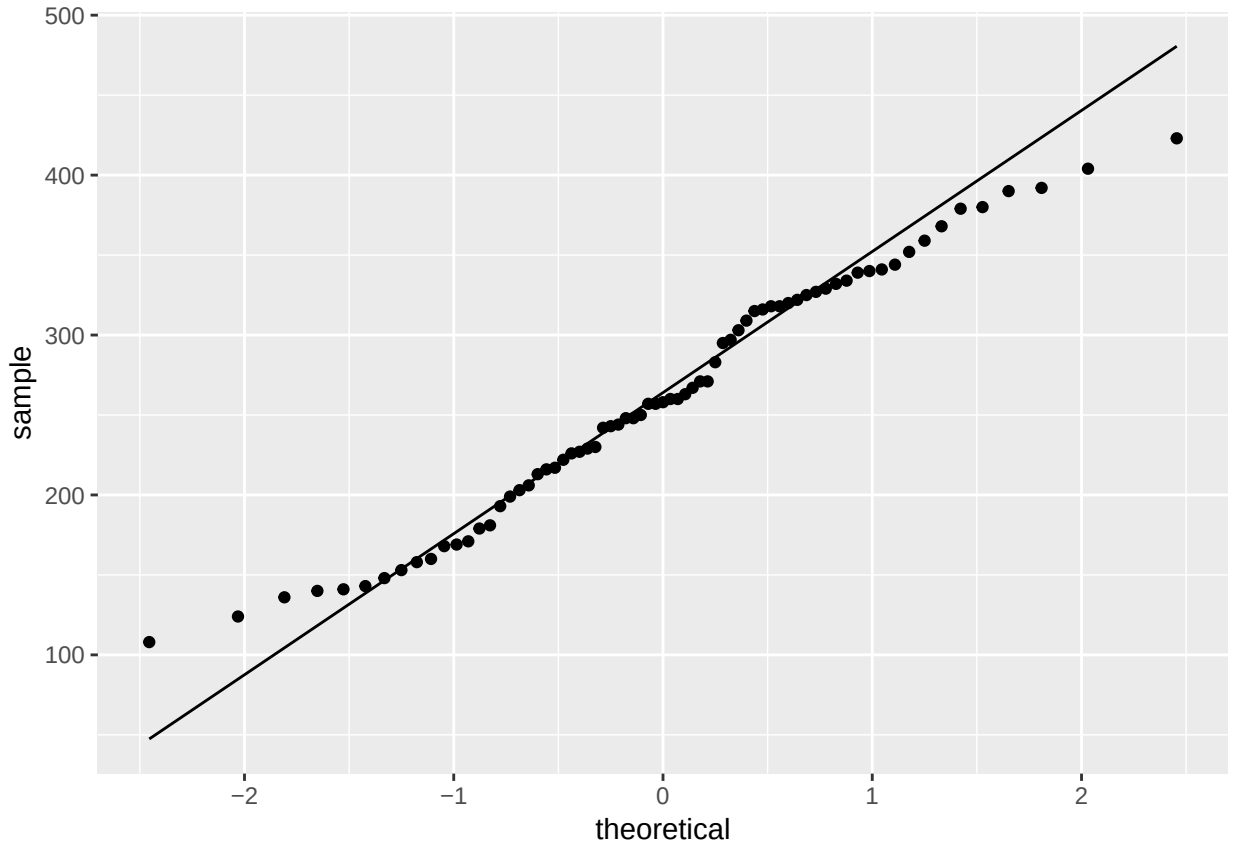
Normal probability plots with ggplot2

Normal probability plots with ggplot2

```
chick_qq <- ggplot(chick_diet, aes(sample = weight)) +
  geom_qq() + geom_qq_line()
```

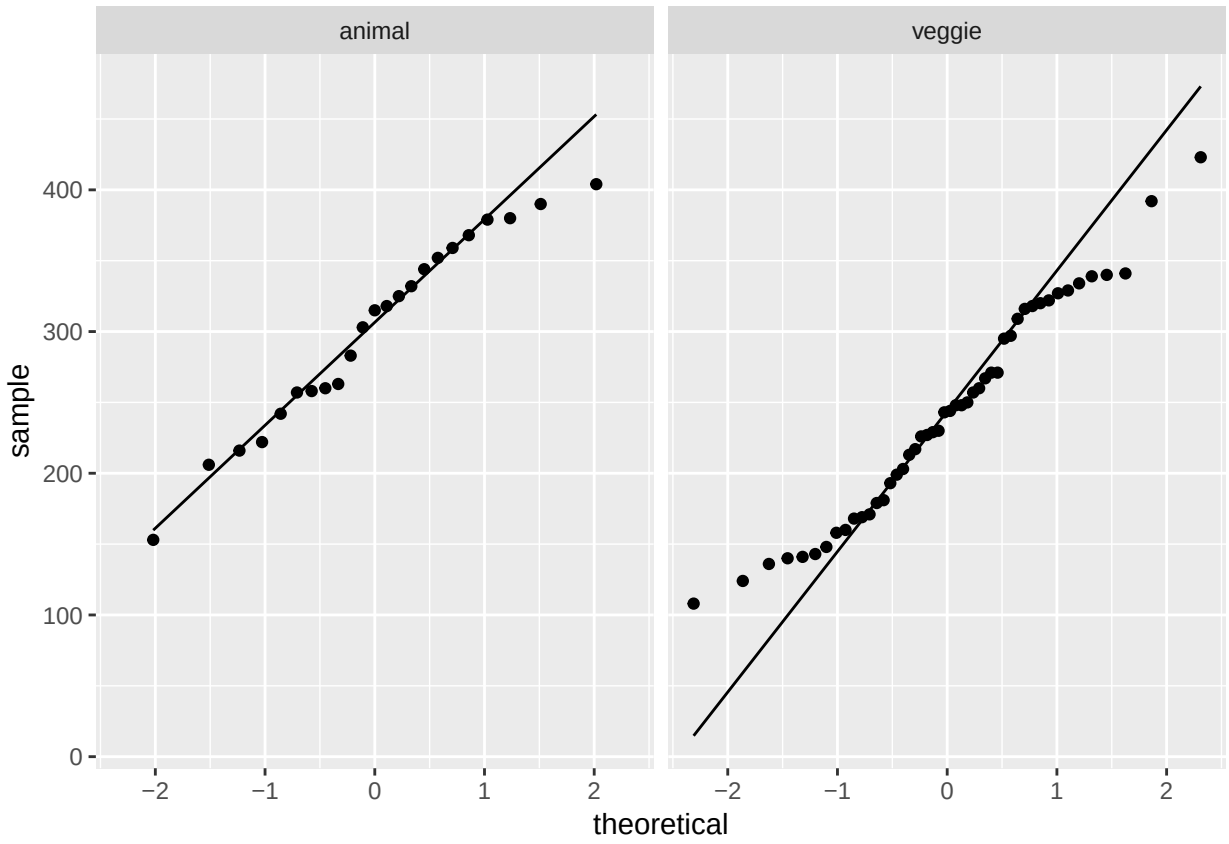
Normal probability plots with ggplot2

```
chick_qq
```



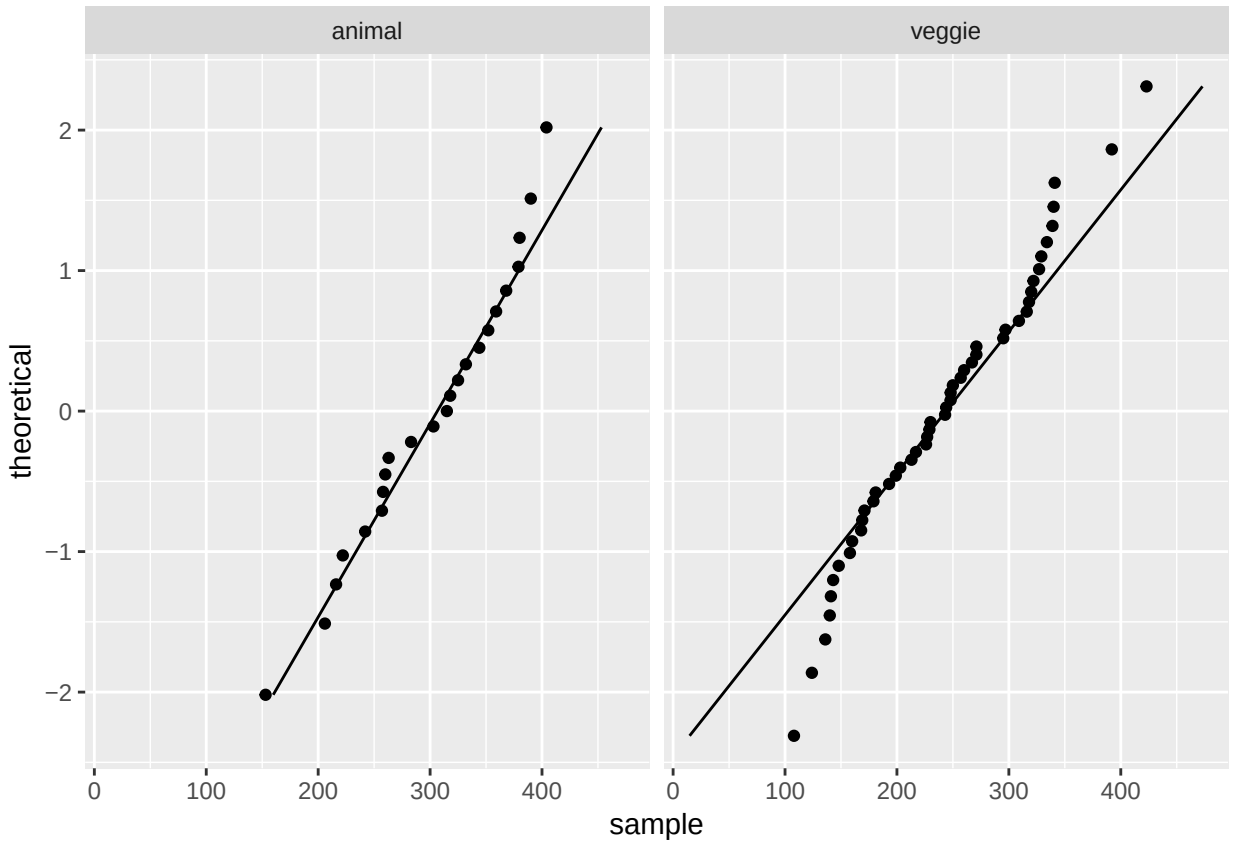
Normal probability plots with ggplot2

```
chick_qq  + facet_wrap("diet")
```



Normal probability plots with ggplot2

```
chick_qq  + facet_wrap("diet") + coord_flip()
```



Normal probability plots with ggplot2

```
chick_qq_complete <- ggplot(chick_diet, aes(sample = weight)) +
  geom_qq() + geom_qq_line() + facet_wrap("diet") + coord_flip()
```

Non-parametric tests (part 1 of 2)

Non-parametric tests (part 1 of 2)

Mann-Whitney U Test: the non-parametric equivalent of an independent two-sample t-test!

```
# t.test(weight~diet, data = chick_diet)
wilcox.test(weight~diet, data = chick_diet)
```

```
## Warning in wilcox.test.default(x = c(325, 257, 303, 315, 380, 153, 263, :
## cannot compute exact p-value with ties
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  weight by diet
## W = 793.5, p-value = 0.003064
## alternative hypothesis: true location shift is not equal to 0
```

Putting it all together

Putting it all together

```
titanic <- read.csv("titanic.csv")
head(titanic)
```

```
##   pclass survived Residence                              name age sibsp
## 1      3     died         0                 Abbing, Mr. Anthony  42     0
## 2      3     died         0      Abbott, Master. Eugene Joseph  13     0
## 3      3     died         0        Abbott, Mr. Rossmore Edward  16     1
## 4      3 survived         0 Abbott, Mrs. Stanton (Rosa Hunt)  35     1
## 5      3 survived         2        Abelseth, Miss. Karen Marie  16     0
## 6      3 survived         0      Abelseth, Mr. Olaus Jorgensen  25     0
##   parch    ticket  fare cabin embarked boat body              home.dest
## 1     0 C.A. 5547  7.55  <NA>        S <NA>   NA                   <NA>
## 2     2 C.A. 2673 20.25  <NA>        S <NA>   NA    East Providence, RI
## 3     1 C.A. 2673 20.25  <NA>        S <NA>  190    East Providence, RI
## 4     1 C.A. 2673 20.25  <NA>        S    A   NA    East Providence, RI
## 5     0    348125  7.65  <NA>        S   16   NA Norway Los Angeles, CA
## 6     0    348122  7.65 F G63        S    A   NA     Perkins County, SD
##   Gender
## 1      M
## 2      M
## 3      M
## 4      F
## 5      F
## 6      M
```

Putting it all together

1. Test the fare of two groups (survivors and non-survivors) for normality

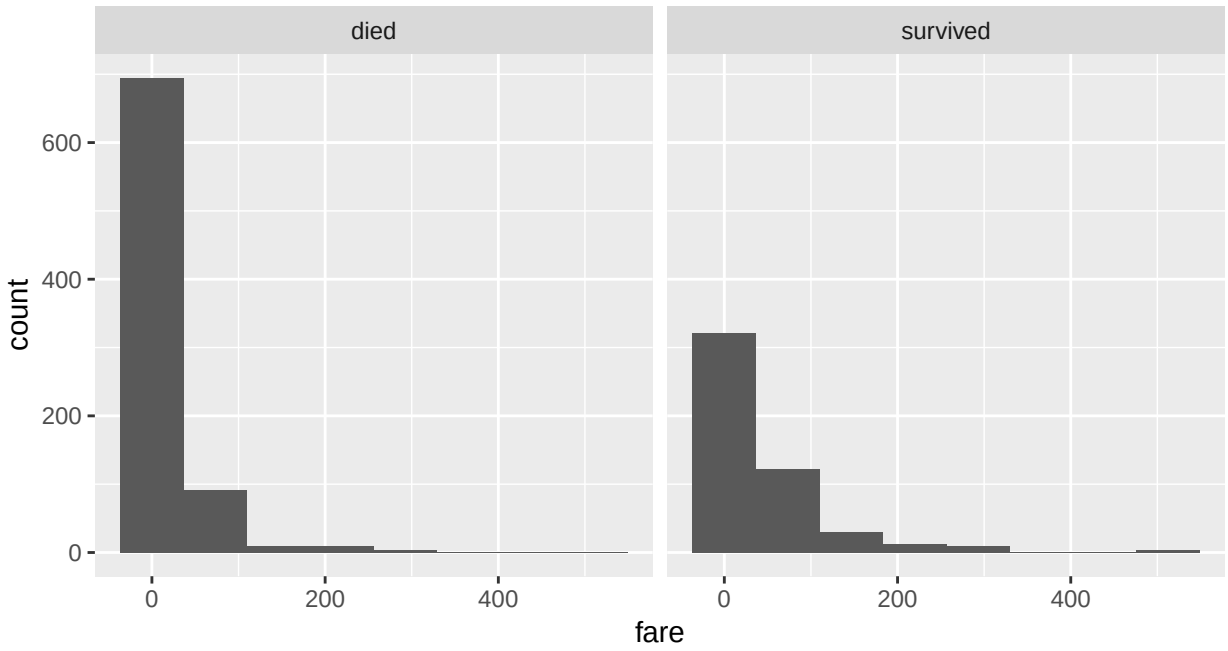
a. Histogram (hint:`facet_wrap()`)
b. Normal probability plot
c. Shapiro-Wilk normality test
d. Levene's test for equal variance

2. Make any data transformations necessary
3. Do a t-test on the transformed data
4. Compare the results with a Mann-Whitney U test

Putting it all together

1a. Histogram

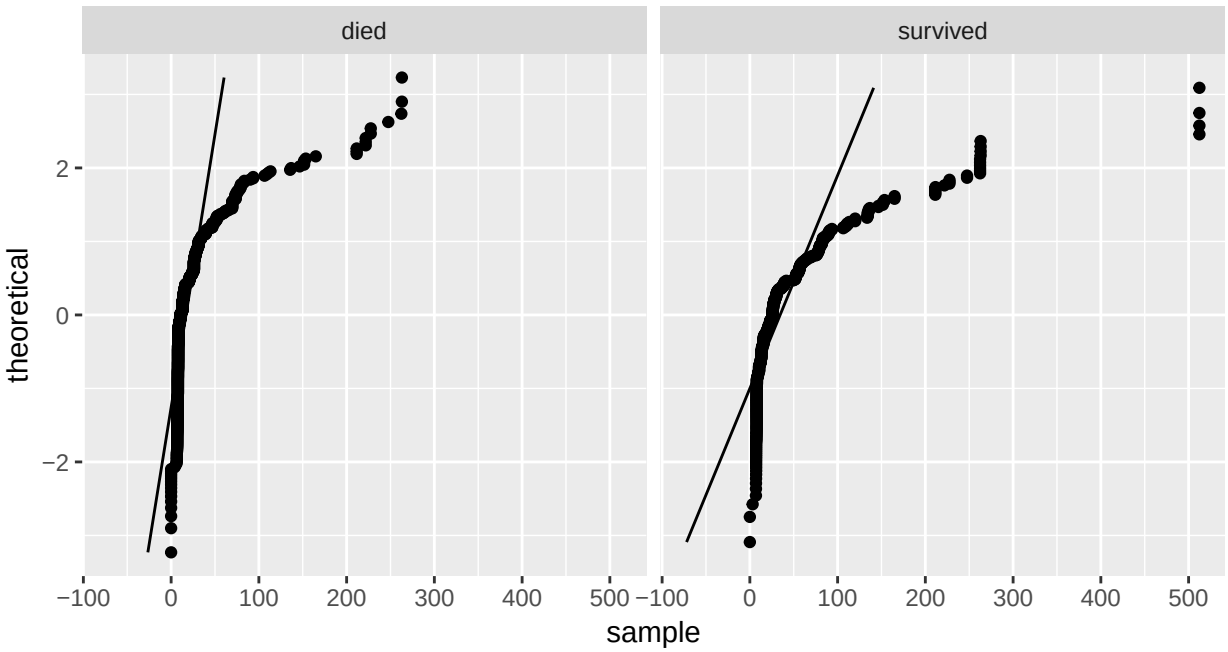
```
ggplot(titanic, aes(x = fare)) + geom_histogram(bins=8) +
  facet_wrap("survived")
```



Putting it all together

1b. Normal probability plot

```
ggplot(titanic, aes(sample = fare)) + geom_qq() + geom_qq_line() +
  facet_wrap("survived") + coord_flip()
```



Putting it all together

1c. Shapiro-Wilk normality test

First, split data by survival

```
titanic_dead <- titanic %>% filter(survived == "died")
titanic_alive <- titanic %>% filter(survived == "survived")
```

Putting it all together

1c. Shapiro-Wilk normality test

```
shapiro.test(titanic_dead$fare)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  titanic_dead$fare
## W = 0.50634, p-value < 2.2e-16
```

```
shapiro.test(titanic_alive$fare)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  titanic_alive$fare
## W = 0.60752, p-value < 2.2e-16
```

Putting it all together

1d. Levene's test for equal variance

```
leveneTest(fare~survived, data = titanic)
```

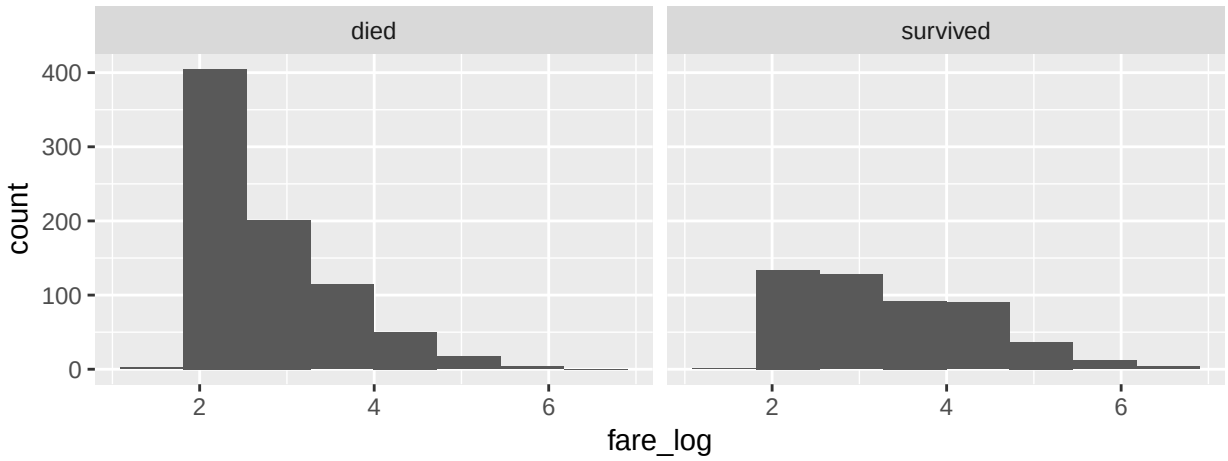
```
## Levene's Test for Homogeneity of Variance (center = median)
##         Df F value    Pr(>F)
## group    1   62.35 6.054e-15 ***
##       1306
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Putting it all together

2. Data transformations

```
titanic_transformed <- titanic %>%
  mutate(fare_log = log(fare))
ggplot(titanic_transformed, aes(x = fare_log)) + geom_histogram(bins=8) +
  facet_wrap("survived")
```

```
## Warning: Removed 17 rows containing non-finite values (stat_bin).
```



Putting it all together

2. Data transformations

Can't take the log of 0!

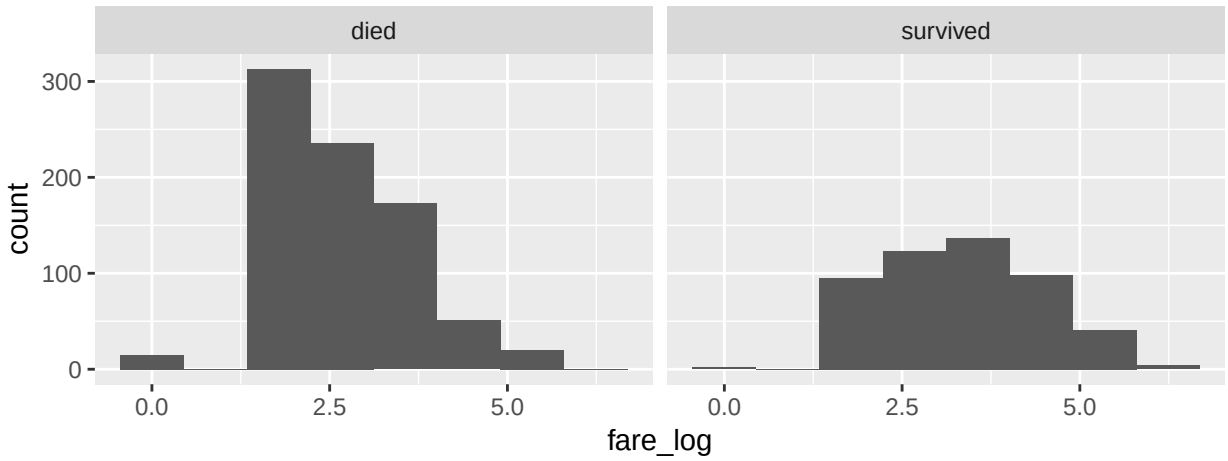
```
titanic_transformed %>% filter(fare == 0) %>% head()
```

```
##   pclass survived Residence                                name age
## 1      1     died         2               Andrews, Mr. Thomas Jr  39
## 2      2     died         2                Campbell, Mr. William  NA
## 3      1     died         2 Chisholm, Mr. Roderick Robert Crispin  NA
## 4      2     died         2        Cunningham, Mr. Alfred Fleming  NA
## 5      2     died         2      Frost, Mr. Anthony Wood "Archie"  NA
## 6      1     died         1                       Fry, Mr. Richard  NA
##   sibsp parch ticket fare cabin embarked boat body
## 1     0     0 112050    0   A36        S <NA>   NA
## 2     0     0 239853    0  <NA>        S <NA>   NA
## 3     0     0 112051    0  <NA>        S <NA>   NA
## 4     0     0 239853    0  <NA>        S <NA>   NA
## 5     0     0 239854    0  <NA>        S <NA>   NA
## 6     0     0 112058    0  B102        S <NA>   NA
##                       home.dest Gender fare_log
## 1                   Belfast, NI      M     -Inf
## 2                       Belfast      M     -Inf
## 3 Liverpool, England / Belfast      M     -Inf
## 4                       Belfast      M     -Inf
## 5                       Belfast      M     -Inf
## 6                          <NA>      M     -Inf
```

Putting it all together

2. Data transformations

```
titanic_transformed <- titanic %>%
  mutate(fare_log = log(fare + 1))
ggplot(titanic_transformed, aes(x = fare_log)) + geom_histogram(bins=8) +
  facet_wrap("survived")
```



Putting it all together

3. Do a t-test on the transformed data

```
t.test(fare_log~survived, data = titanic_transformed)
```

```
##
##  Welch Two Sample t-test
##
## data:  fare_log by survived
## t = -11.119, df = 926.38, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.7143850 -0.5000357
## sample estimates:
##     mean in group died mean in group survived
##               2.747297               3.354507
```

Putting it all together

4. Compare the results with a Mann-Whitney U test

```
# t.test(fare_log~survived, data = titanic_transformed)
wilcox.test(fare~survived, data = titanic)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  fare by survived
## W = 131450, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

Homework time!