Linear Regression - assumptions

NRES 710

Last compiled: 2024-08-19

Review

This week we will continue to explore linear regression by talking about **assumptions of regression** – and in fact these are important assumptions of many other statistical tests. And then we will discuss how to use linear regression models to make **predictions**.

Most important thing we have learned so far:

- $Y_i = \beta_0 + \beta_1 x_i + \epsilon \sim N(0, \sigma)$

There are many assumptions that this regression test makes! But we will focus on what are considered the five most important of these assumptions.

Five Regression Assumptions

What are the assumptions of statistical test and how do they influence your results?

The first thing you need to know about assumptions of statistical tests (regression, t-test, ANOVA, other tests we will cover...) is that the tests are **robust to violations of assumptions**.

- Robust means that: if an assumption is violated, it very rarely influences the results we get from the analysis (e.g., slope).
- We'll talk about which of the assumptions influence different parameters...
- But for most parameters, if the assumptions are violated it does not influence the slope.
- Assumption violation may influence the standard deviation, but this is not often reported.
- **Violations cause the p-value to increase.** This means that violations are likely to be conservative. Since we want to avoid committing Type I error (rejecting the null when in fact it is true), then if assumption violation causes p-values to increase then we are less likely to commit Type I error.

A general rule (*axiom*) in statistics is that: **the more assumptions a test makes, the more powerful it is (power = p-values).**

We often use statistical tests that make assumptions. Often, these assumptions are true. And since we make more assumptions, we are more likely to detect a significant effect. **If these assumptions are valid.**

I often don't care too much about assumptions – because they are often met due to our **study design**, the specific analysis we chose, and **most analyses are robust to violations of assumptions**. But, **reviewers do**. Reviewers will try to find something wrong with your paper. They will try to find something wrong and jam up the process. So it can be useful to carefully document how you examined for violations of assumptions in your analysis. This gets tedious, but is part of the 'statistical ritual' of our field... (Or maybe it shouldn't be. Johnson [1999] made suggestions that our statistical ritual leads to bad practices, and we will read another paper to this effect at the end of the semester.)

But as for me, again, I don't care too much about these assumptions, because **most analyses are robust to violations of assumptions**.

There are **5 assumptions** to linear regression. I put the equation up on the board again because most of these assumptions are indicated in the equation, either explicitly or implicitly.

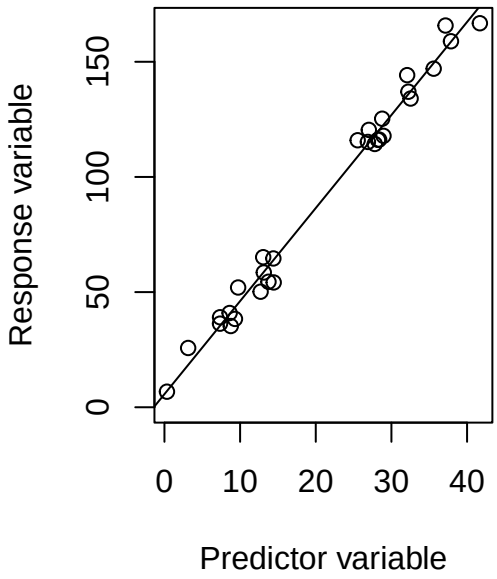
Continuous Y

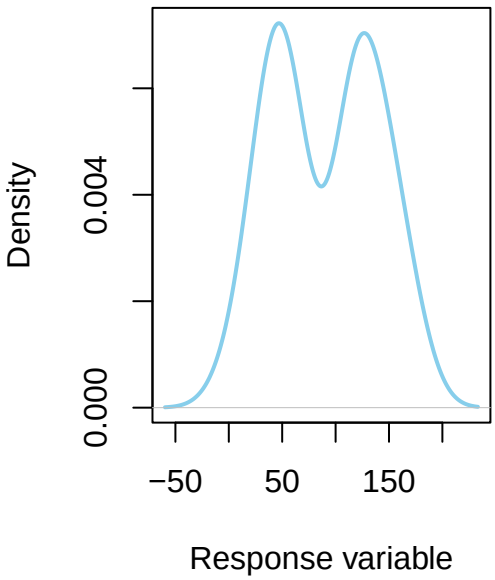
- Should be continuous; not a count, like a number. But... if it is a count, that's potentially okay, because regression/ANOVA are robust to violations of assumptions. We don't really need a test for this – if you collected the data, you should know whether it is continuous or not!
- Note: linear regression also assumes that your X-variable is continuous... but we can relax this assumption, and we will do so next week when we explore t-tests! More on this soon.
- If you Y is not continuous, it will not influence your slope, but it might increase your p-value.

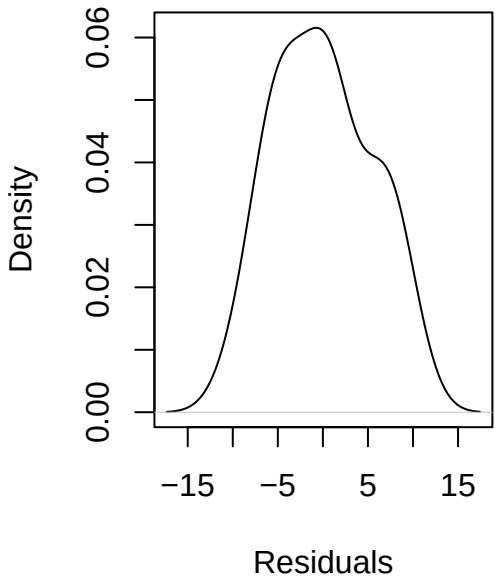
Error is normally distributed

- Very clearly indicated in the equation! Important and something that a lot of people get wrong. Some folks say that X-variable has to be normally distributed, continuous, etc. – but nope. Others assume that your Y-variable has to be normally distributed. Nope! This assumption relates to the **error** around the Y-variable.

For example: consider the data in the left graph. There is a gap in values for the middle-range of X-values. If we examine this as a frequency histogram (middle graph), this is not a normally-distributed y-variable; it is bimodally distributed! This is okay. When we run this regression, we have not violated any assumptions, because the error is normally distributed around the line.







Then why are people always asking if the Y-variable is normally distributed. . . ? This is because if you look at the distribution of the y-data and it appears normal, the residuals will almost always be normal when you run the analysis.

But, if you run a histogram and your y-data are not normal, this does not necessarily mean that your error will also not be normal. To really know if the assumption is violated, you need to run the regression and examine whether the residuals are normal (third graph).

If your error is not normally distributed, it's not going to influence your slope – but it might increase your p-values a little bit.

Linear relationship between X and Y

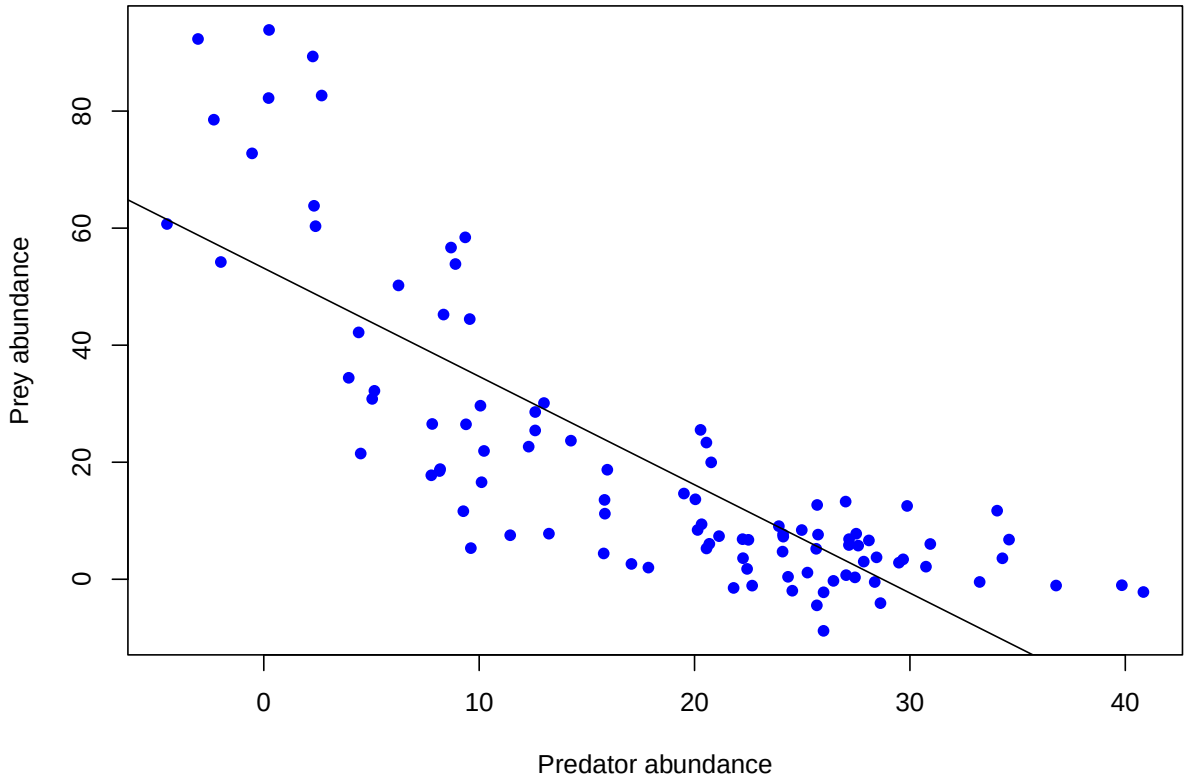
This one is important. This is implicit in our linear model equation.

It assumes that there is a single parameter – the slope – describing the relationship between X and Y.

The reason why I think this is important is because so often in ecology/natural resource management I see people obtaining two continuous variables and immediately running a regression model – without ever considering whether their data have a linear relationship. They don't even think about it.

This is a problem, because in ecology. . . many processes of interest are non-linear!

Example: mesocosm experiment. We want to understand the effect of crayfish predators on prey fish. Does fish abundance decrease and predatory crayfish increases?



This very clearly violates the assumption of linearity! We can fit a better statistical model – one that does not assume linearity – which can better help us measure this relationship and explain it to the scientific community!

So, don't assume there is a linear relationship between X and Y – examine this, verify, and adjust your model as needed.

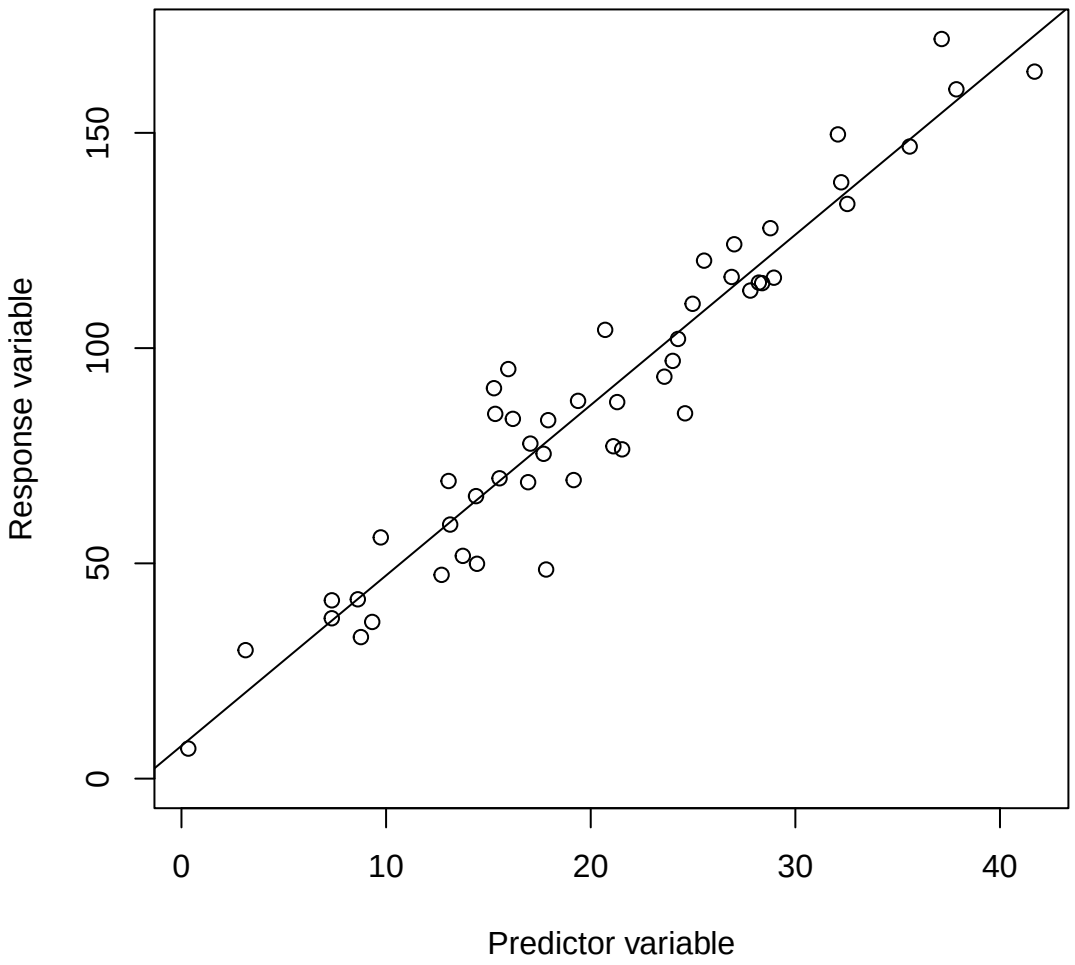
If the relationship is not linear, **this will alter your slope**! We are measuring something that is not linear an

Homoscedasticity

homo = same; *scedasticity* = variance, noise, error, etc.

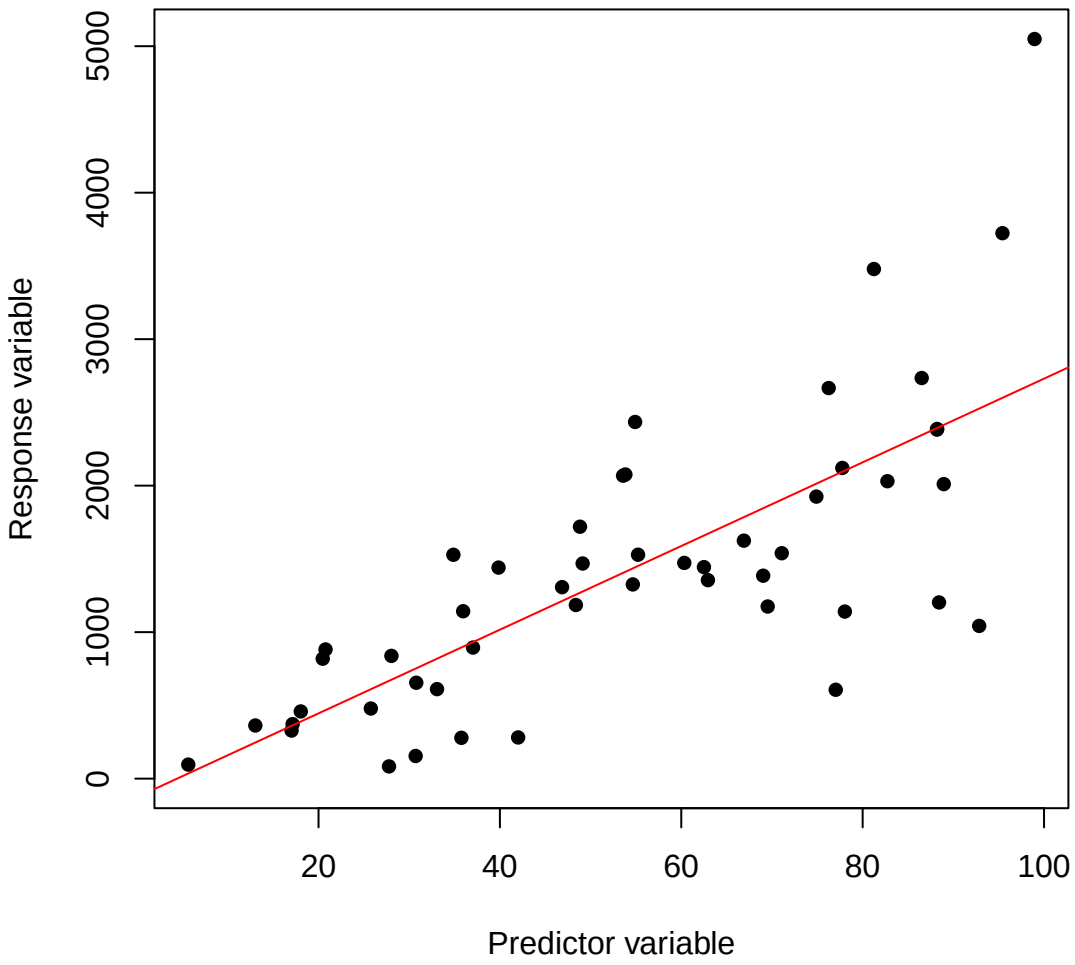
Implicit in our equation: - $\epsilon \sim N(0, \sigma)$

Constant variance: *error does not change not matter what the X and Y values are.*



We can visualize homoscedasticity by imagining/drawing normally distributed bell curves amidst our data along the regression line...

An example of **heteroscedasticity** is any case where your variance changes. This commonly occurs in ecology when we count animals. For example, when we are electofishing for fish in a river, areas with no fish have little variance; areas with many fish have high variance! It creates this cone-shaped data.



We can visualize *heteroscedasticity* by imagining/drawing normally distributed bell curves amidst our data along the regression line, and the bell curves get wider as we increase along X.

Heteroscedasticity could also happen in a non-linear way.

Q: Will heteroscedasticity influence your slope? No.

We can account for *heteroscedasticity* in our model by adding a weighting paramter to the error component: $\epsilon \sim N(0, \sigma * y)$ would allow for error to increase with y!

Independent samples

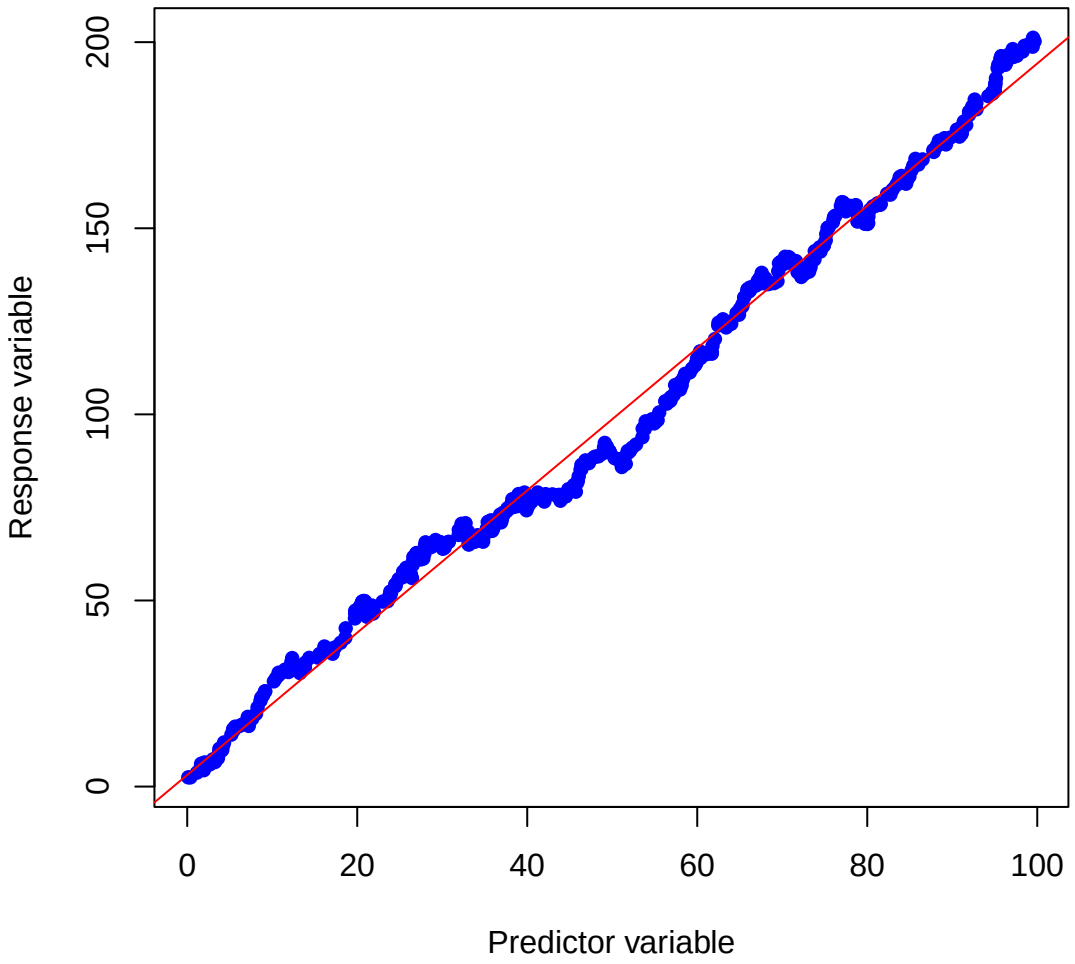
When people say that the x-variable is the 'independent variable', this is what they really mean! Your samples should be independent of eachother and there should be **no autocorrelation**.

Example: measuring pollution in water samples every 10 m down the middle of a river from a chemical plant. These data will have autocorrelation, because pollution can't change that much from sample to sample.

Q: How might you eliminate or minimize autocorrelation? Maybe by increasing distance between samples – measure every kilometer. This decreases autocorrelation, but also decreases sample size. Tradeoff...

I personally don't worry too much about autocorrelation, because often when you fix it/account for it, nothing changes. If you had a strong slope and p-value with one test, you will likely get a strong slope and p-value with another test.

Consider these autocorrelated data:



Instead of our points bouncing around the line, they tend to follow eachother.

This will not affect our slope, and it probably won't affect the p-value too much (would only increase). So again, linear regression is robust to violation of this assumption.

Two types of autocorrelation issues: spatial and temporal autocorrelation. We will discuss how to deal with this down the road. But again, it's not too big of a concern.

Many things are autocorrelated in nature. Animal movements, for example! Is this a problem?

Maybe not. This is what animals do – they move! Try to get big sample sizes.

Evaluating assumptions w/ graphs

Statistical tests exist to statistically test for these assumptions. These are p-value generating tests. There are some consequences of this.

- If you have a small sample size, the assumption will never be violated! Because of the relationship between sample size and p-values that we have identified in previous classes.
- Conversely, if you have a really large sample, the assumption will always be violated!

So, for these reasons, I don't like these tests, and I don't recommend using these tests.

Instead, what I do I look at my data graphically! And I will teach you to do this also. We will visually examine our data to identify whether our data meet these assumptions or not. If it has been violated, we will see this. If we can see the assumption has been violated, then we now know this.

Useful rule of thumb:

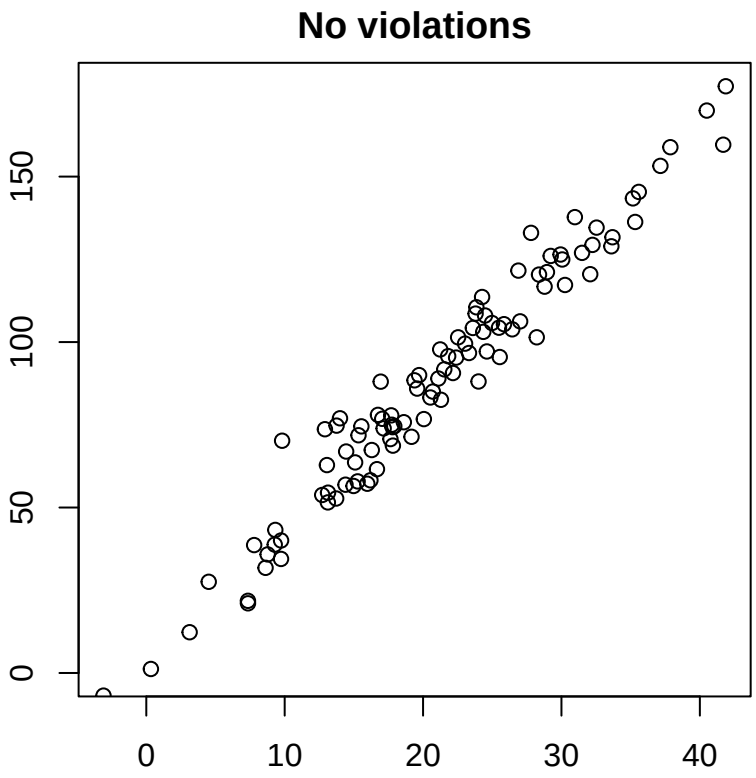
- If you can't see it, it doesn't exist, and you assume there isn't one.
- If you can see it, then an assumption may be violated, and then it's our decision to do something about it or not.

We have four main assumptions, and we will use four graphical approaches to examine whether these assumptions are met.

Assumption	X-Y Scatterplot	Residuals Scatterplot	Histogram of Residuals	Autocorrelation Function (ACF)
Normality				
Linearity				
Homoscedasticity				
No autocorrelation				

X-Y Scatterplots

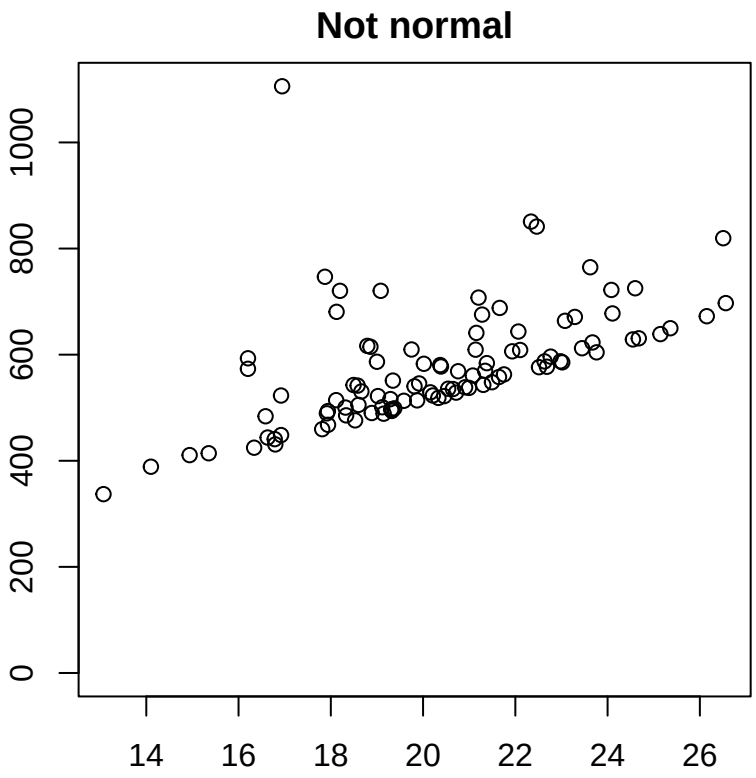
This first graph shows data where **none of the assumptions are violated**.



- Data are very clearly linear.
- Data appear to be normally distributed around the line. Most are close to the line, but some are out in the tails.
- Does not appear to be any autocorrelation.
- Does not appear to be any heteroscedasticity.

This is how your data should look!

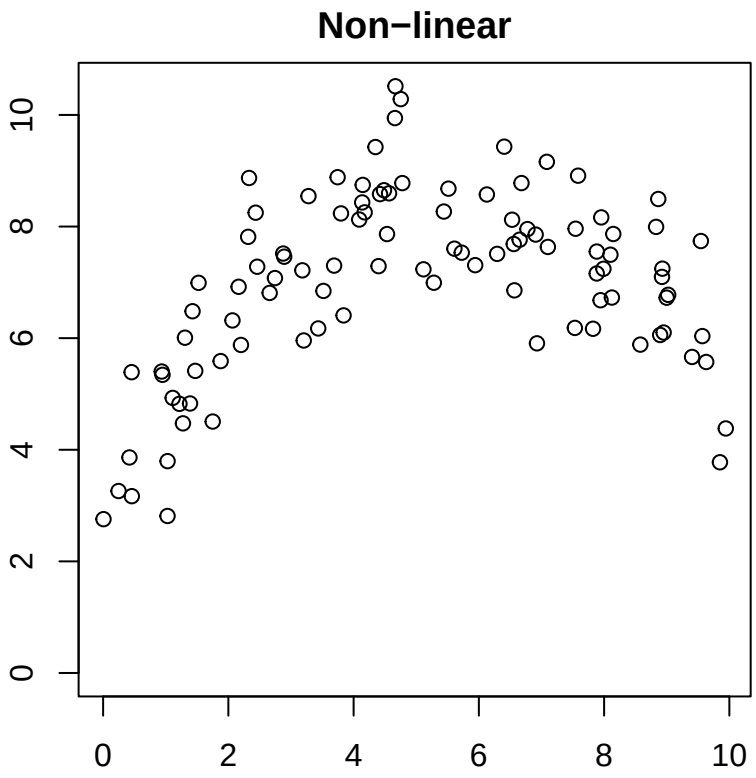
This second graph shows data were the assumption of error normality is violated.



We can see that normality is violated because there are no tails below the line!

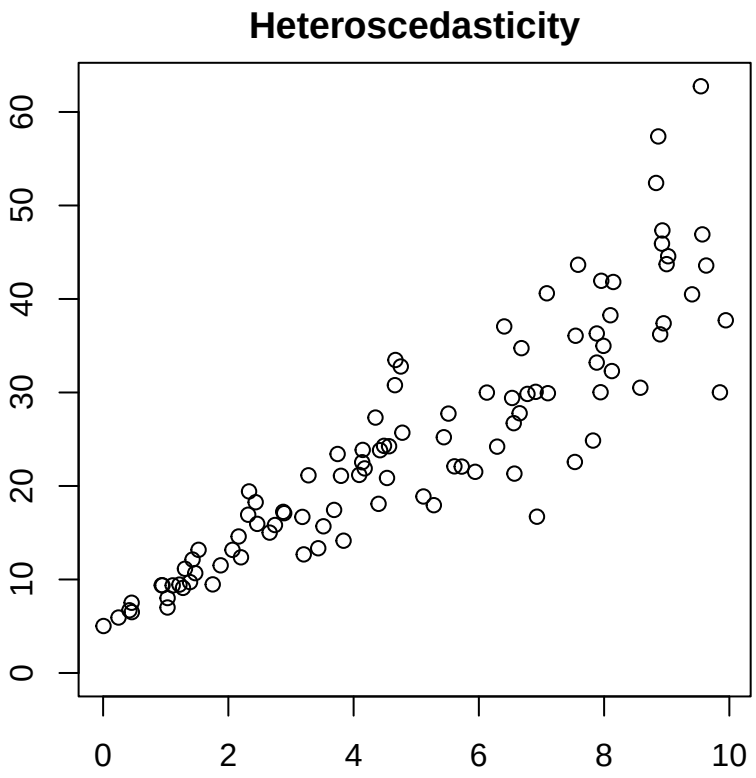
Q: Has this influenced the slope? No. It would influence the intercept (but nobody reports that ever, so no big deal).

This third graph shows data where the assumption of **linearity** is violated.



- A linear regression model would not fit these data well, so we would want to seek an alternative approach.
- Note: Some might say that these data are not normally distributed, but it is. Most points are centered on the line, with some out on the tails.

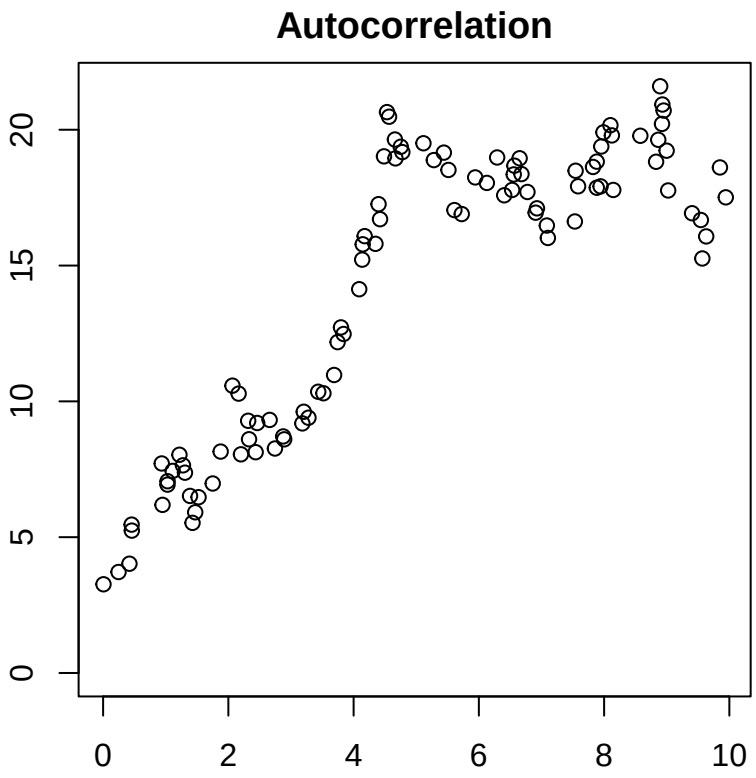
This fourth graph shows data where the assumption of **homoscedasticity** is violated.



These data are **heteroscedastic**; as X increases, error increases.

We don't need a statistical test to know that these data are heteroscedastic.

This last graph shows when the data are not independent and autocorrelation is present in the data.



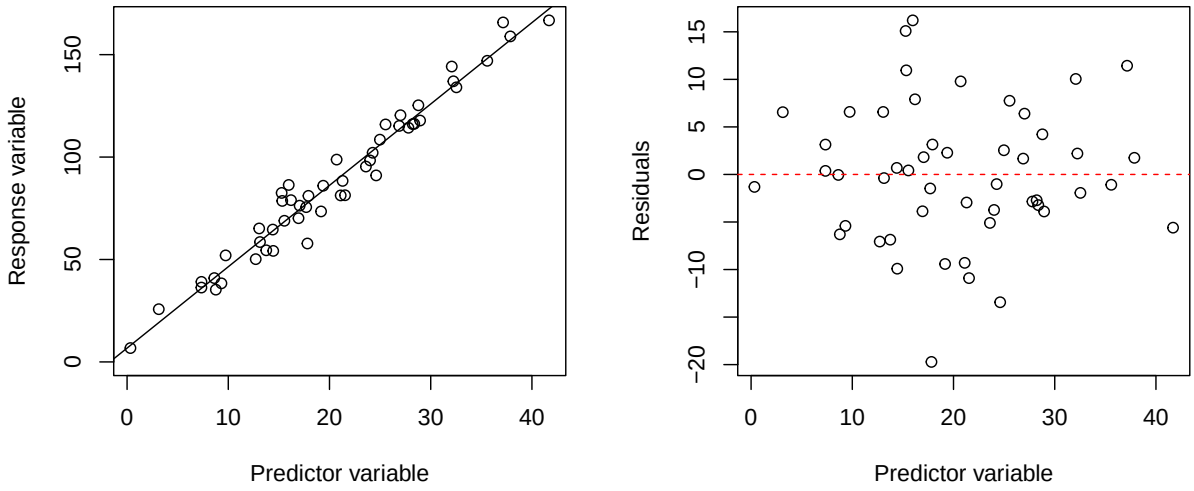
What do we see here. . . ?

- The errors are similar to each other; i.e., they are correlated to one another.
- The error is not centered on the line, but rather *follows itself.*
- Two types of autocorrelation; we'll discuss this in a future lecture when discuss how to fix or model autocorrelation (which requires more complicated models, no ready for this just yet).

Residuals Scatterplot

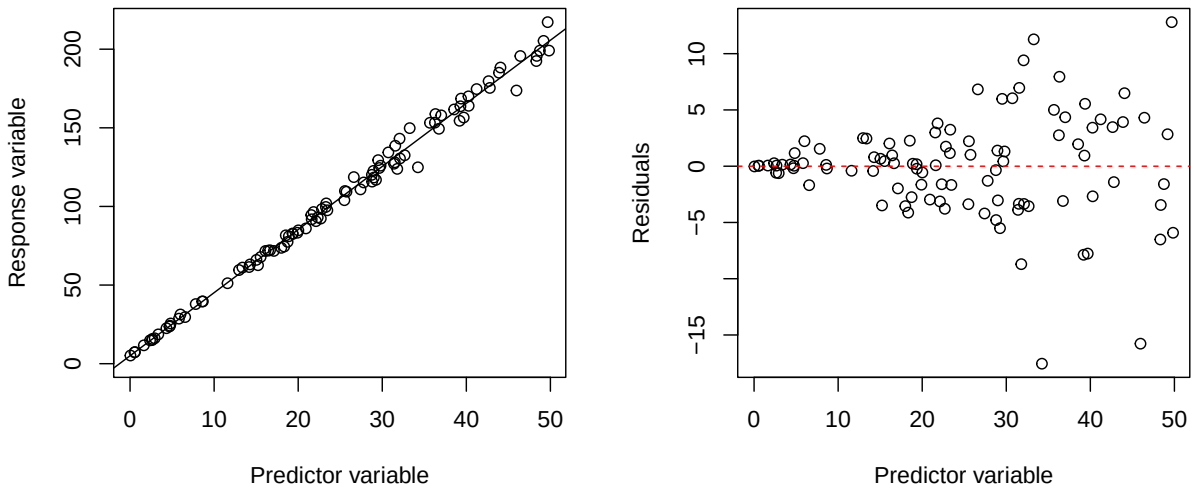
Residuals scatterplot involve making a graph of:

- **X-variable –> X data**
- **Y-variable –> residual (error)**



Assuming we had a pretty usual data with X and Y and we fit a regression line (above), we can revisualize that graph with the same X-variable but now with the residuals of Y on the y-axis. We sort of flip that graph, make the regression line be at 0, and then residuals above and below that line are visualized with the Y-variable. This is a 'residuals plot'.

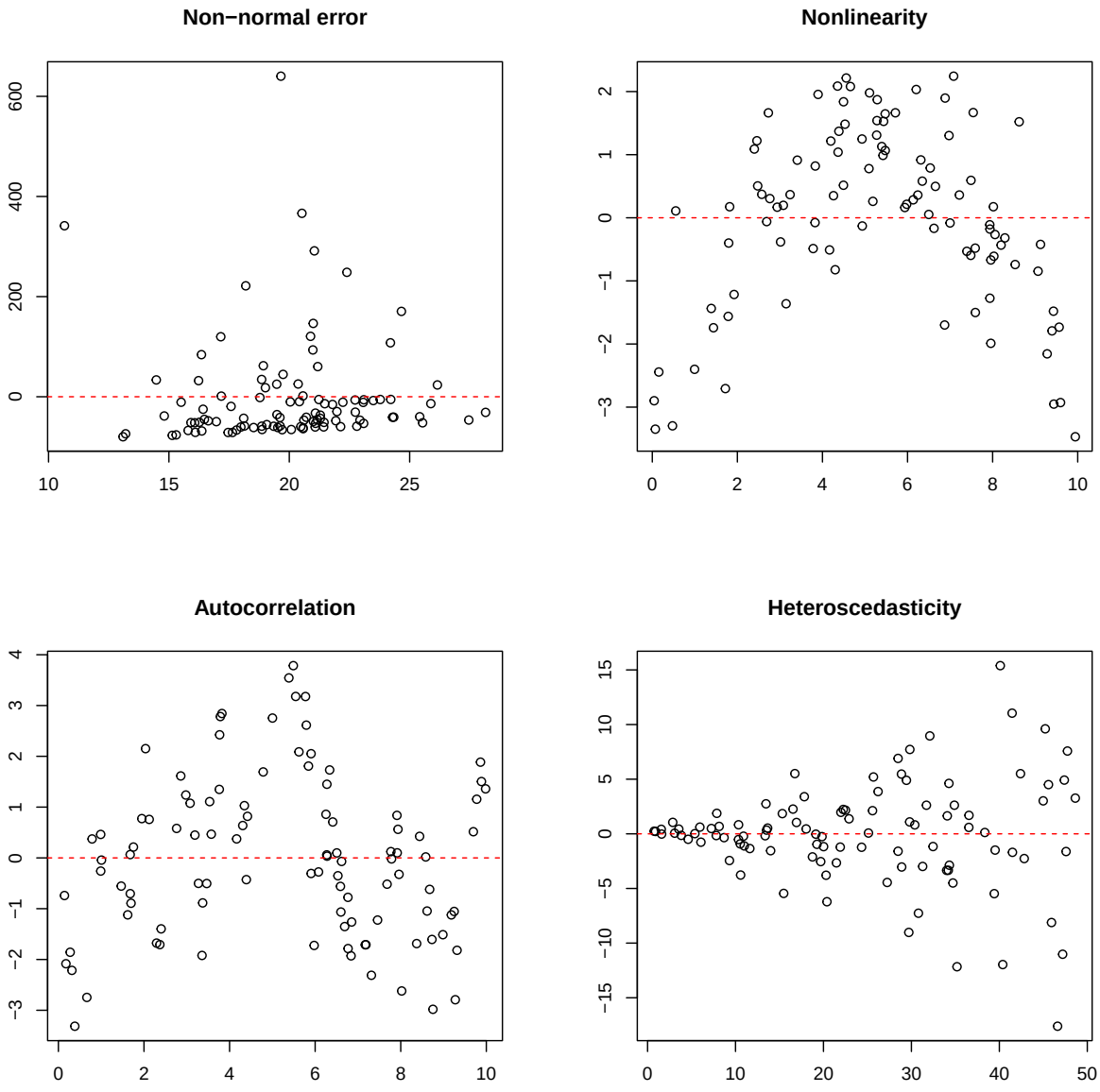
For example, the residuals plot can be useful when you have a large range of X and Y, and your error is very small:



The data look pretty tight around the line in the X-Y scatterplot, but when we look at the residual plot, we see something else. This is common when we have low error – small noise.

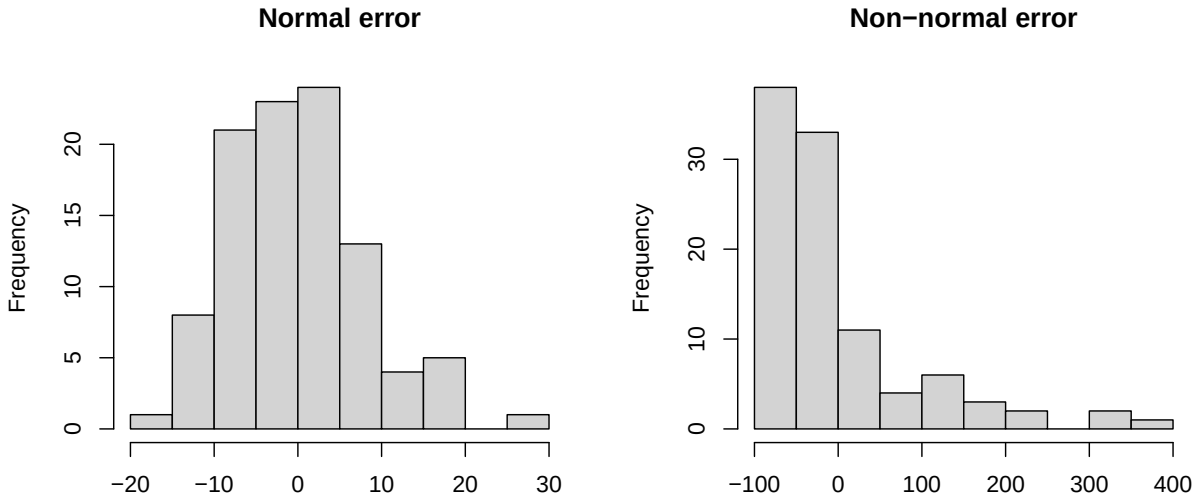
Q: What is happening here – what assumption has been violated?

Residuals scatterplots are useful for all four of these assumptions. Here are the four simulated datasets we used for X-Y scatterplots but not visualized using residual scatterplots:



Histogram of Residuals

This is a way to look at the residuals from a global perspective, so it is most useful for looking at normality. Not useful for the others. You can see heteroscedasticity with it (kirtosis), but most useful for checking for normality.



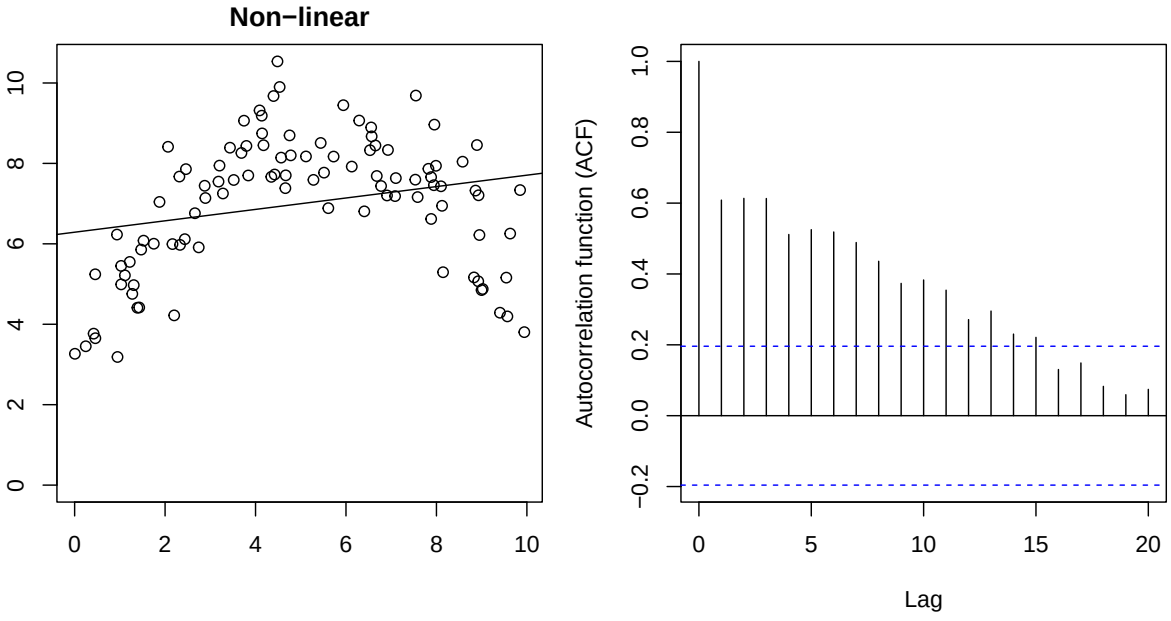
Autocorrelation Function (ACF)

The autocorrelation function (**ACF**) shows us the correlation for the residuals. It teaches us about the probability that:

- If one point is above the line, what is the chance that the next point will also be above the line?
- Alternatively, if another point is below the line, what is the chance that the next point will also be below the line?

As you might expect, it is best for teaching us whether our data are **autocorrelated**. But it can also tell us if our data may be **nonlinear**. A quick example:

For example:



So, if we use an ACF and it shows autocorrelation, we should make sure we don't have a non-linearity issue.

Summary

Assumption	X-Y Scatterplot	Residuals Scatterplot	Histogram of Residuals	Autocorrelation Function (ACF)
Normality	X	X	!!	
Linearity	X	X		**
Homoscedasticity	X	X		
No autocorrelation	X	X		!!

Graphical Assumption Tests

We described graphical approaches to evaluating whether your data meet assumptions of linear regression. We discussed five assumptions:

- **Continuous Y** – your Y-variable is continuous; only an eye-ball test is needed for this
- **Linearity** – a linear relationship between X and Y
- **Normality residuals** – residuals (error) are normally distributed
- **Homoscedasticity** – constant variance; noise around regression is constant across all values of X
- **No autocorrelation** – your data are independent of eachother

We can use different graphs to test these assumptions (**see above table**).

I am going to move to R and we will learn how to easily make these plots in R!

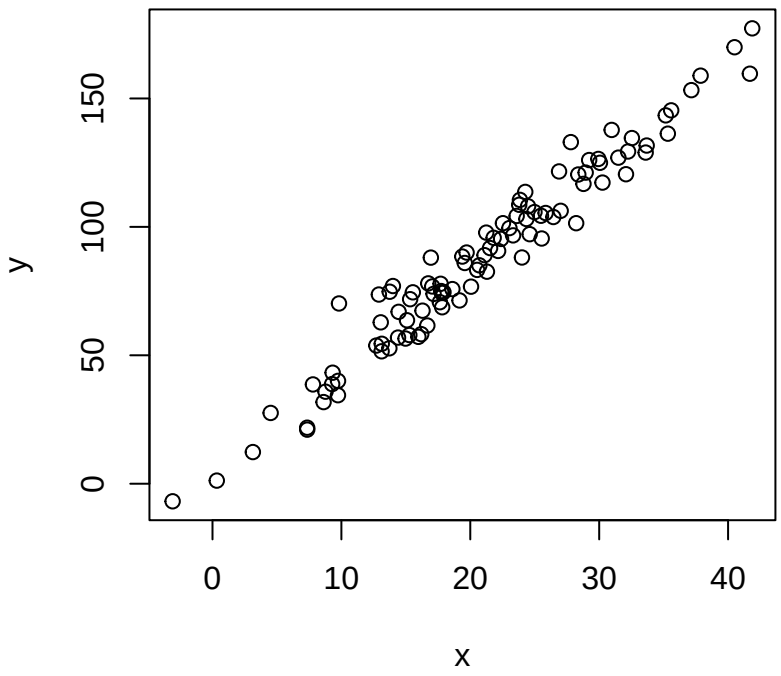
Save code and data for this lecture in your working directory:

- Click here to download code. This is general code to help make these graphs.
- Click on these links to download and save the data: good data, nonlinear data, nonnormal data, heteroscedastic data, and autocorrelated data.
  - Note: all of these datafiles were simulated using code at the end of this lecture; see that code if you are interested in understanding 'truth'.

X-Y Scatterplots

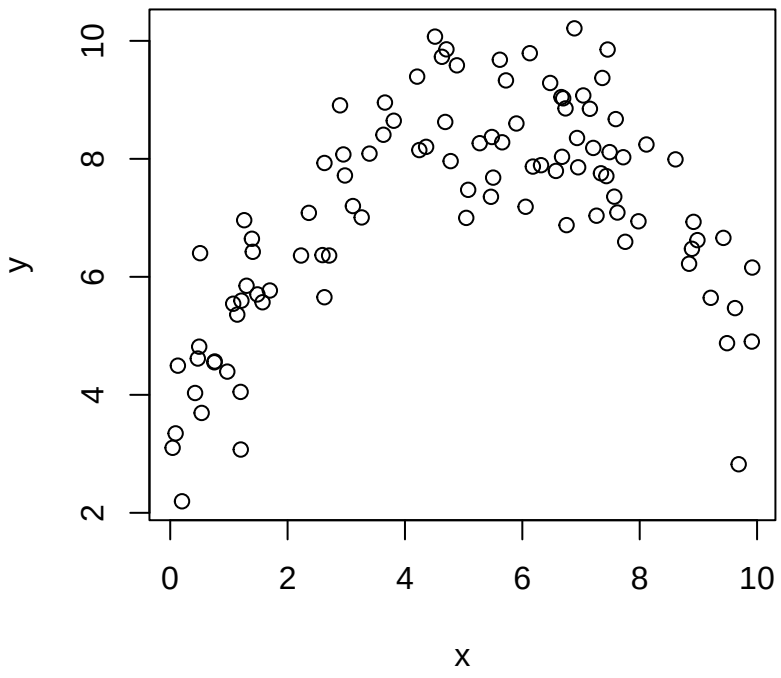
Let's start by examining a dataset that has no issues with it!

```
# Load the data
datum <- read.csv("lecture_6_good_data.csv")

# Plot the data
plot(y ~ x, data = datum)
```



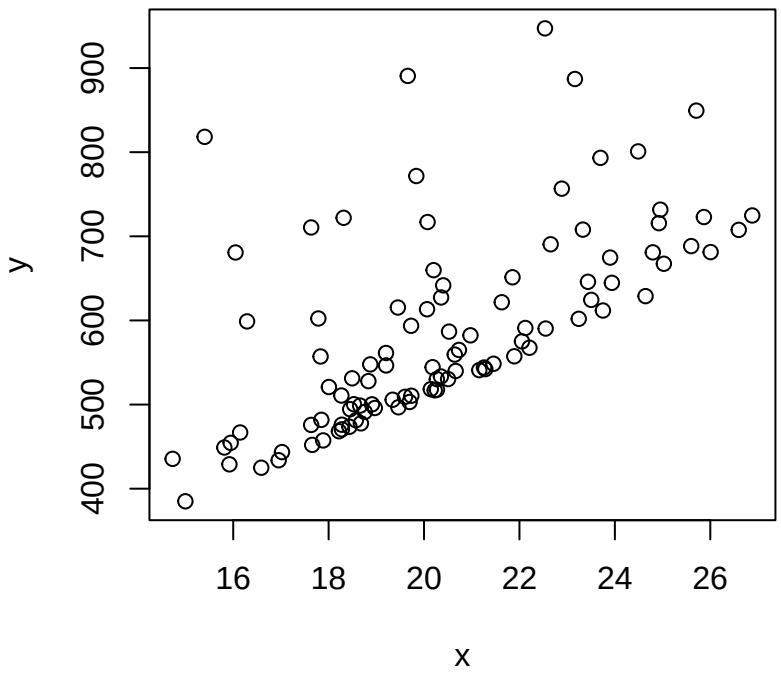
Y is continuous, definitely linear, the residuals appear normally distributed (most close to the line), variance seems constant, does not appear to have any autocorrelation (but it's possible).

```
# Load the data
nonlinear <- read.csv("lecture_6_nonlinear_data.csv")

# Plot the data
plot(y ~ x, data = nonlinear)
```



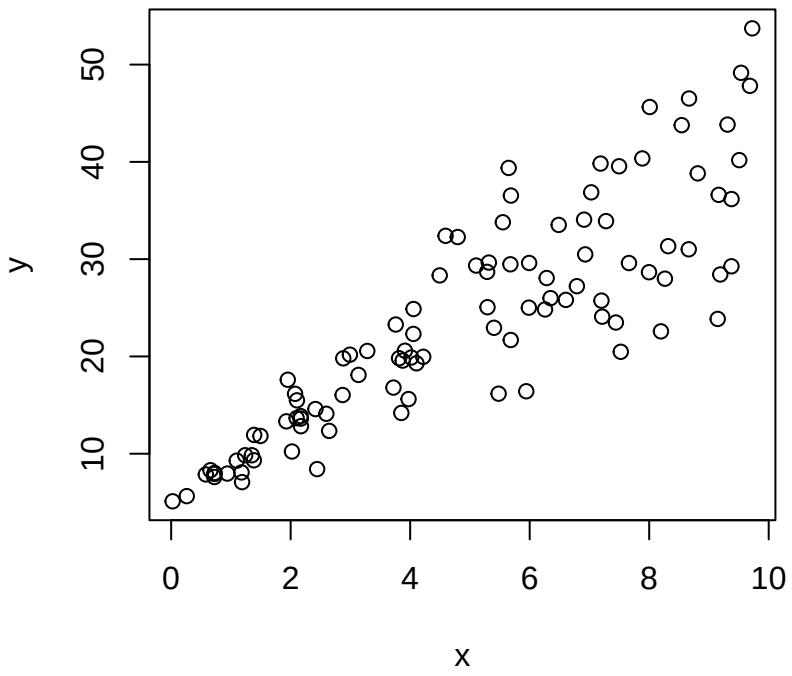
Very clearly nonlinear!

```
# Load the data
norm <- read.csv("lecture_6_nonnormal_data.csv")

# Plot the data
plot(y ~ x, data = norm)
```



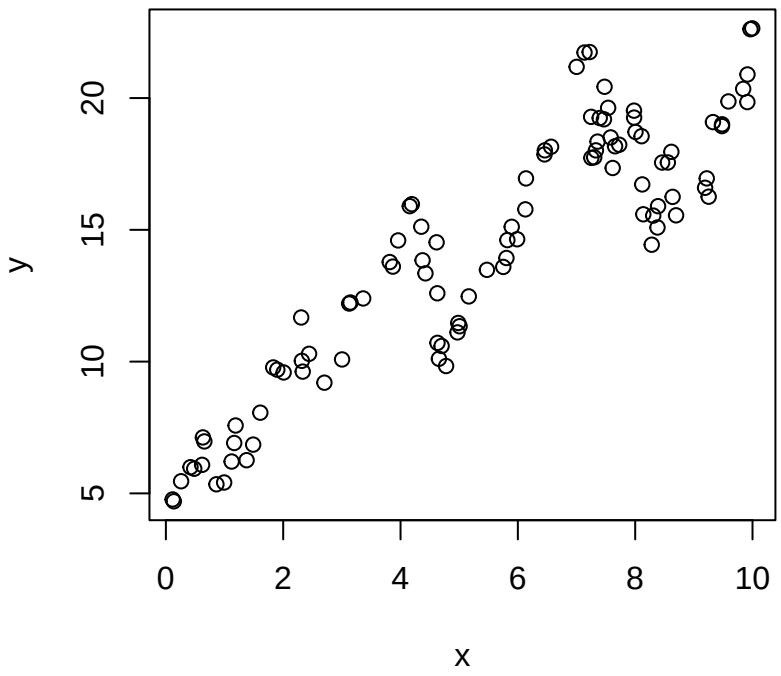
These data have a heavy tail about the line, but don't really have any tail below the line. The residuals are likely not normally distributed.

```
# Load the data
hetero <- read.csv("lecture_6_heteroscedastic_data.csv")

# Plot the data
plot(y ~ x, data = hetero)
```



As the X-variable increases, noise/error in Y increases. This is clearly heteroscedastic.

```
# Load the data
auto <- read.csv("lecture_6_autocorrelated_data.csv")

# Plot the data
plot(y ~ x, data = auto)
```



Pretty easy to see the autocorrelation in these data. Most observations of Y at X are similar to the observation of Y at X-1. **The data follow eachother!** But, if autocorrelation isn't strong, it can be difficult to see.

Residual Plots

Before we can run a residual plots, we have to generate the residuals! Which means we have to first fit a regression. X-Y Scatterplots use the raw data, but the other three graphs we use require a regression to be run for us to then make these plots. Let's run a few analyses:

```
# Fit linear regression models to the five datasets
results <- lm(y ~ x, data = datum)
resultsNonlinear <- lm(y ~ x, data = nonlinear)
resultsNorm <- lm(y ~ x, data = norm)
resultsHetero <- lm(y ~ x, data = hetero)
resultsAuto <- lm(y ~ x, data = auto)
```

I'm not going to look up the summaries for each of the models. We could use these to look at summaries and maybe infer how much violations influence slopes, p-values, etc. It's tricky to compare these different datasets because I simulated them all using different slopes, intercepts, etc. So we can't quite compare them directly. However, we have already talked about those 'rules of thumb' and instead we should just keep them in mind.

The code to extract residuals is called 'residuals()'. This tells us the distance from each point in our dataset to the line of best fit identified by the regression analysis. E.g.,

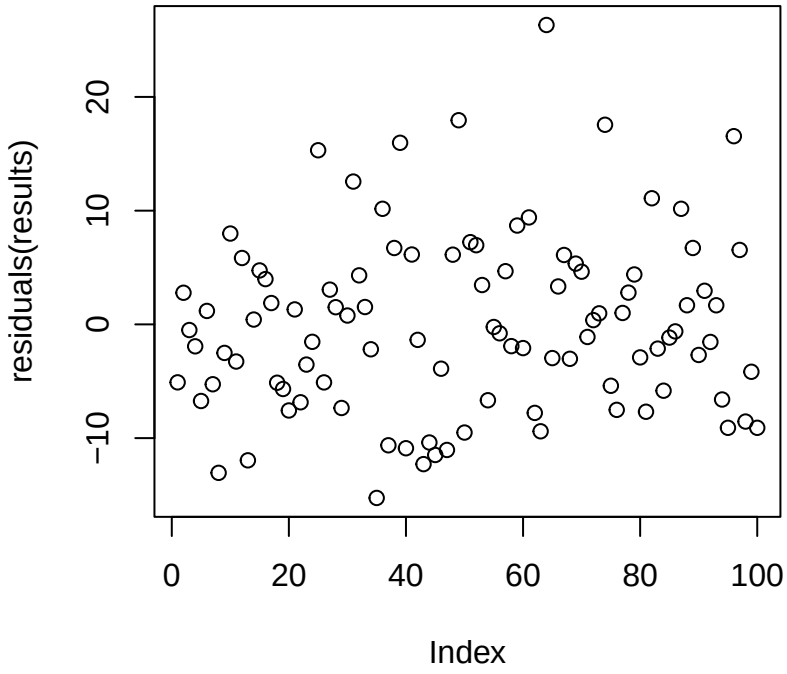
```r
# Examine the residuals (just the first ~20)
residuals(results)[1:20]
```

```
##       1       2       3       4       5       6       7       8       9      10      11
## -5.0961  2.7809 -0.4968 -1.9283 -6.7363  1.1821 -5.2633 -13.0521 -2.5077  7.9873 -3.2665
##      12      13      14      15      16      17      18      19      20
##  5.8372 -11.9524  0.4244  4.7444  3.9817  1.8768 -5.1288 -5.6808 -7.5691
```

These data are used to calculate the **standard deviation of the error** around our line – simply by calculating the standard deviation of these data. The mean of these data should be ~ 0.

But we want to examine these residuals graphically. It may be tempting to plot these residuals very simply using:

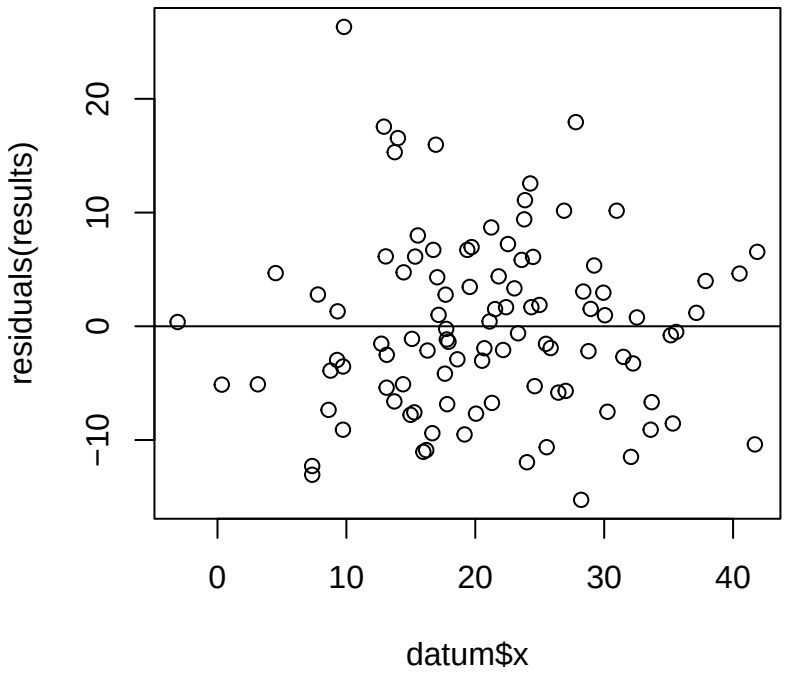
```r
# Simple way
plot(residuals(results))
```



However, note that the X-axis title is "Index". This means that the X-values here are in the order of the data, from 1 to n – the sample size. We actually want to make sure that the X-axis in this plot is our actual X-variable, so we can look at how residuals change along the continuous nature of our X-variable.

We are drawing the residuals from the 'results' object, so we can't reference the 'datum' object like we usually do. Instead we have to explicitly call the raw X-variable data. We can call an individual variable from an object in R using the money sign command: 'datum$x'. We can call'. E.g.,

```r
# Correct way to examine residuals plot
plot(residuals(results) ~ datum$x)

# Add a horizontal line at y = 0
abline(a = 0, b = 0)
```



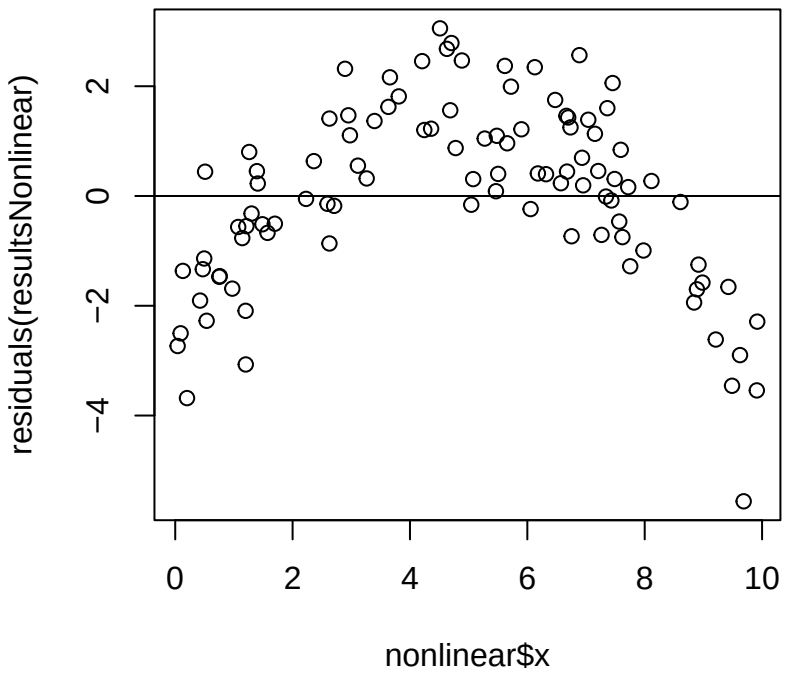
So what have we done. . . ? We have taken our scatterplot of data and:

- flattened it out so that it now fills the whole plot,
- the regression line is now also flattened and at y = 0,
- visualized the degree to which each point is above or below that line.

We can see that (1) most of our residual points are close to the line, few are far away, (2) there's no heteroscedasticity, (3) doesn't appear to be any autocorrelation, and things appear linear.

Let's examine this for nonlinear data:

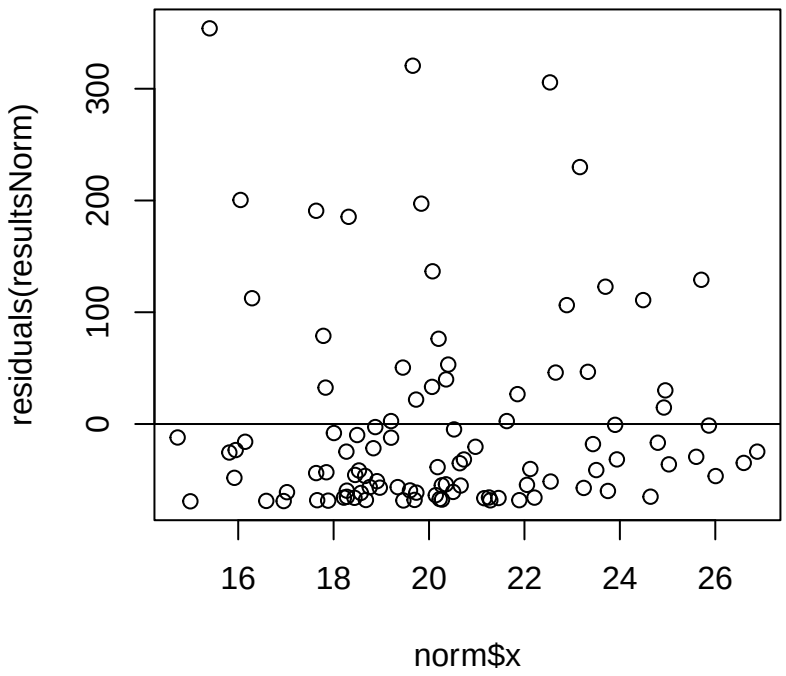
```r
# Correct way to examine residuals plot
plot(residuals(resultsNonlinear) ~ nonlinear$x)

# Add a horizontal line at y = 0
abline(a = 0, b = 0)
```



Clearly this is nonlinear: bunch of points below the line, then a bunch above, and then more below. This would be much better fit with a curvilinear line, which we will discuss in a couple weeks.

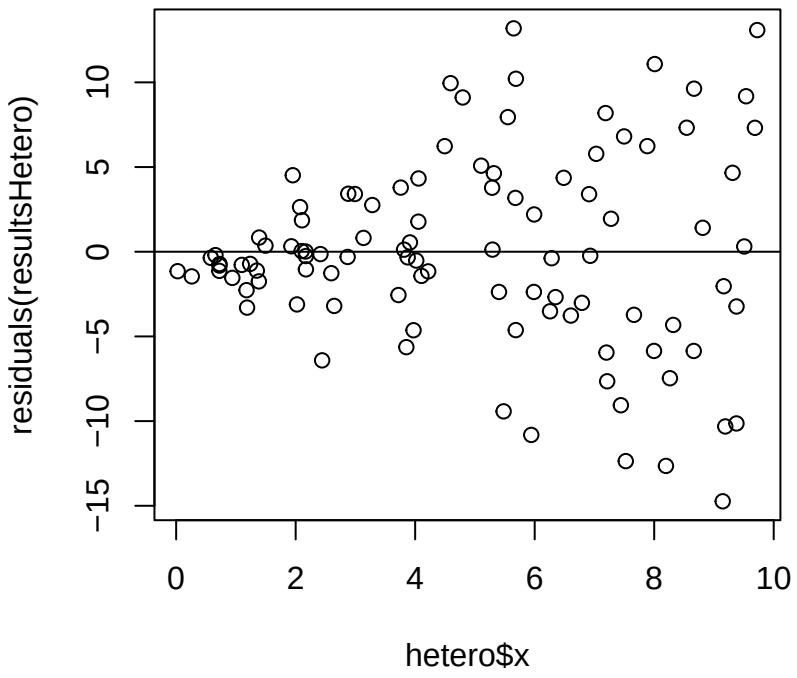
Non-normal data:

```
# Correct way to examine residuals plot
plot(residuals(resultsNorm) ~ norm$x)

# Add a horizontal line at y = 0
abline(a = 0, b = 0)
```



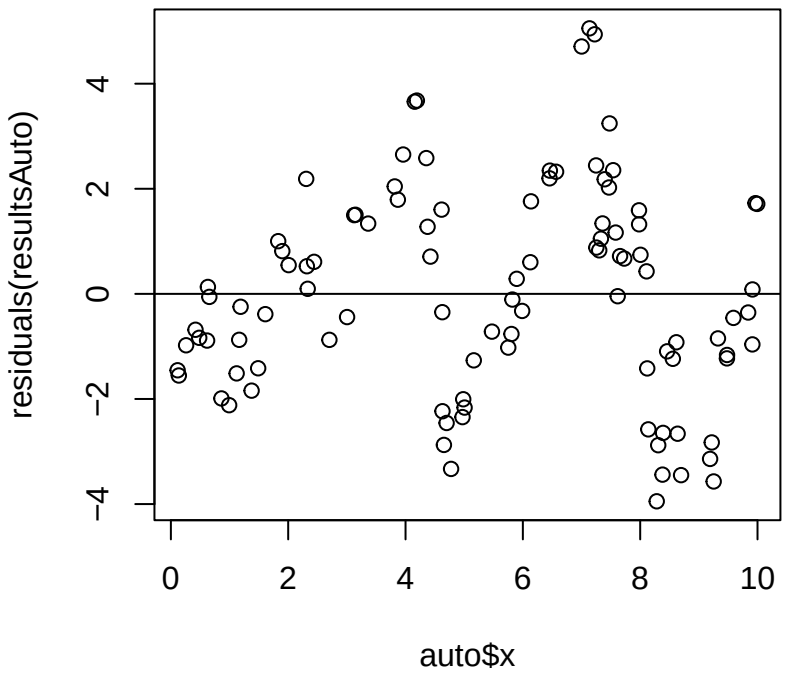
Notice where the regression line is! It's way down at the bottom of the graph, rather than being in the middle. This is a red flag. And then most of the noise is above the line. This suggests non-normality.

```
# Correct way to examine residuals plot
plot(residuals(resultsHetero) ~ hetero$x)

# Add a horizontal line at y = 0
abline(a = 0, b = 0)
```



Pretty easy to see the heteroscedasticity: small variance with small X-values, and then larger variance with larger X-values.

```
# Correct way to examine residuals plot
plot(residuals(resultsAuto) ~ auto$x)

# Add a horizontal line at y = 0
abline(a = 0, b = 0)
```

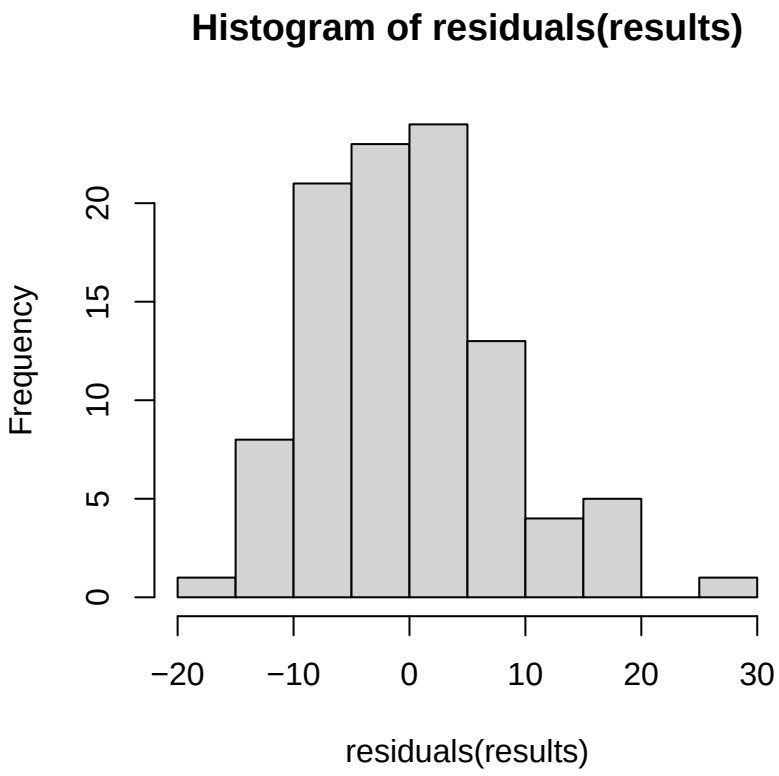


The points (residual error) are following each other! This path is a good indication of autocorrelation. The data are not independent of one another.

Histogram of residuals

The histogram of residuals will make a simple bar chart that examines '*global*' normality across all of your data. What I mean by 'global' is that it does examine normality as the X-variable increases, but lumps all the error across all X-values into a single chart.

```
# Correct way to examine residuals plot
hist(residuals(results))
```



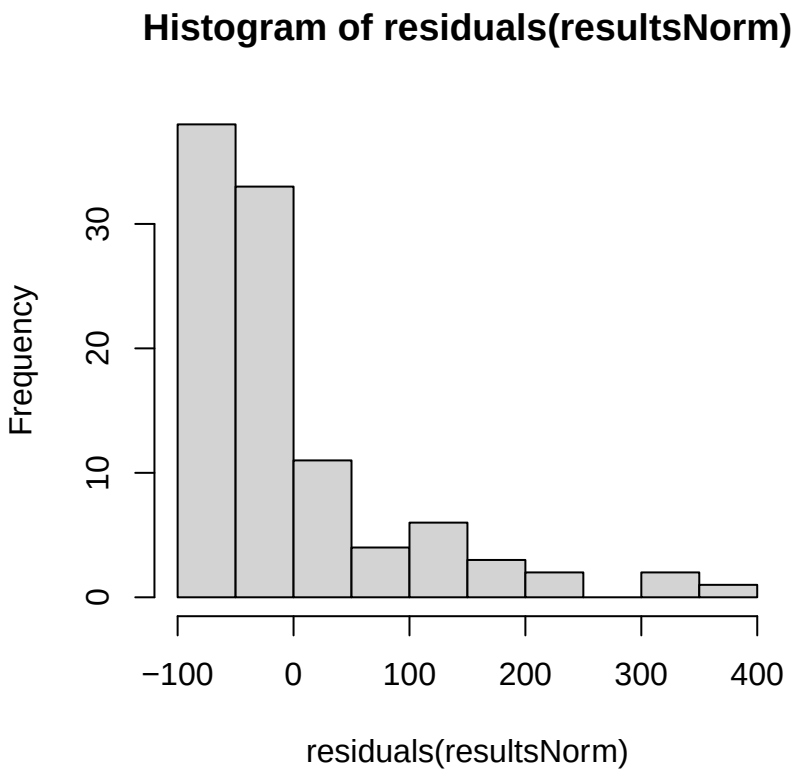
We have taken all of the residuals and examined the distribution of residuals across all the X-values.

Q: Can anyone think of a negative consequence of lumping in this histogram? Can you think of another assumption that we can we no longer test for with this graph?

Q: Does this look perfectly normal?

Nah, not really. It never will. But what we are looking for is obvious, egregious violations of the assumption of normal error. For example, let's make this graph for the non-normal data:

```
# Correct way to examine residuals plot
hist(residuals(resultsNorm))
```

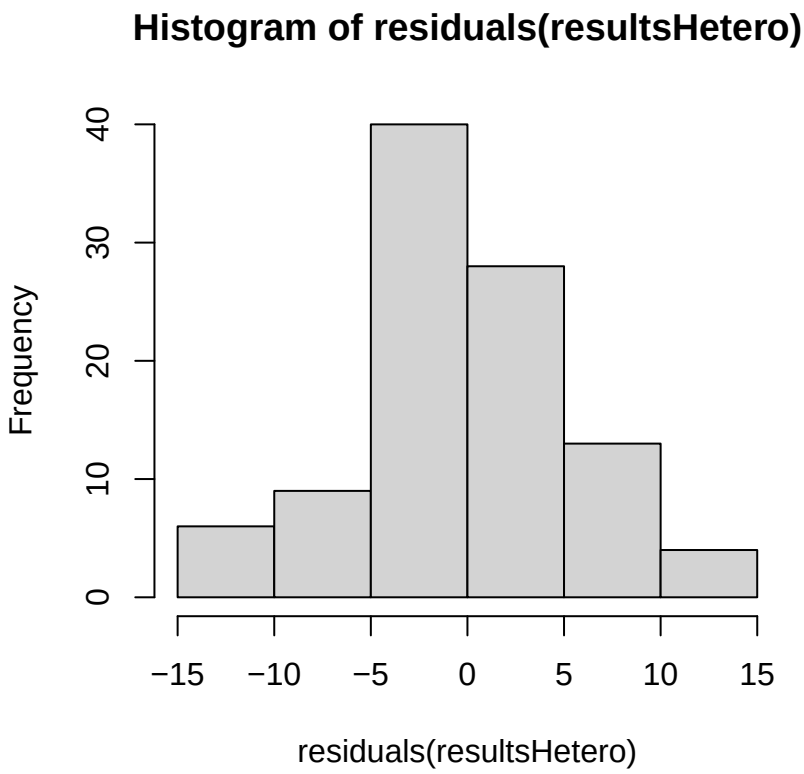


This is pretty bad! This is a case where we would say "Our data are not normal!"

This might be a time where you would use a 'log transformation' to fix this issue. I don't really like log-transformations because it screws with how we interpret our results – makes the story less clear – but that would be an option here.

Let's examine the heteroscedastic data:

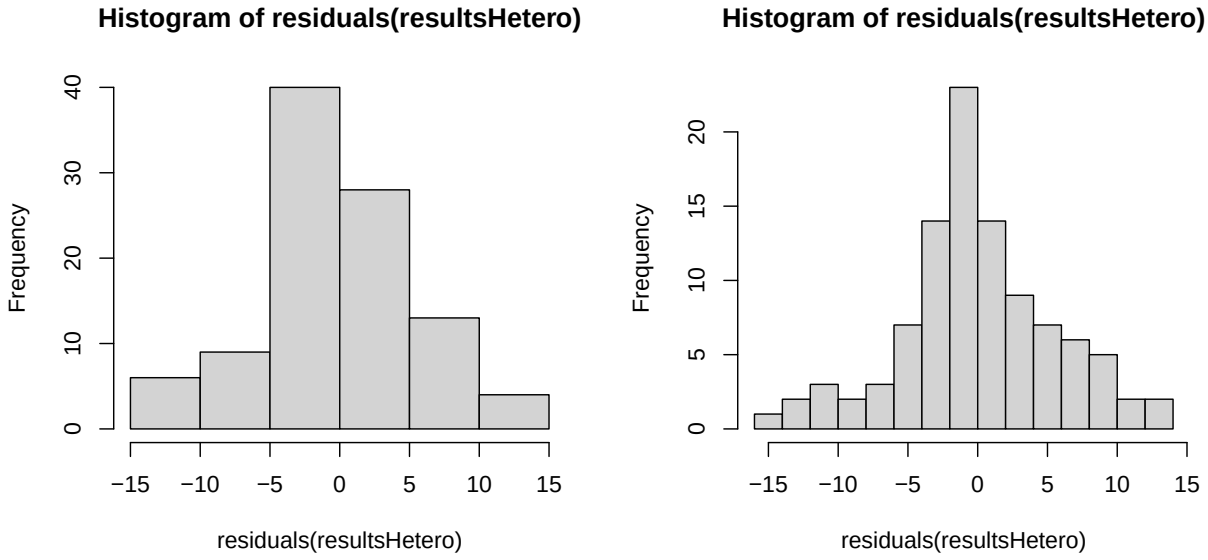
```
# Correct way to examine residuals plot
hist(residuals(resultsHetero))
```



Q: Does anything seem off about these data? (Kind of hard to say.)

If you don't like the appearance of your histogram and want to adjust it, you can change the appearance of it using the 'breaks' command:

```
# Make two graphs side-by-side
par(mfrow = c(1, 2))

# Default setting
hist(residuals(resultsHetero))

# Revisualizing with more breaks
hist(residuals(resultsHetero), breaks = 10)
```



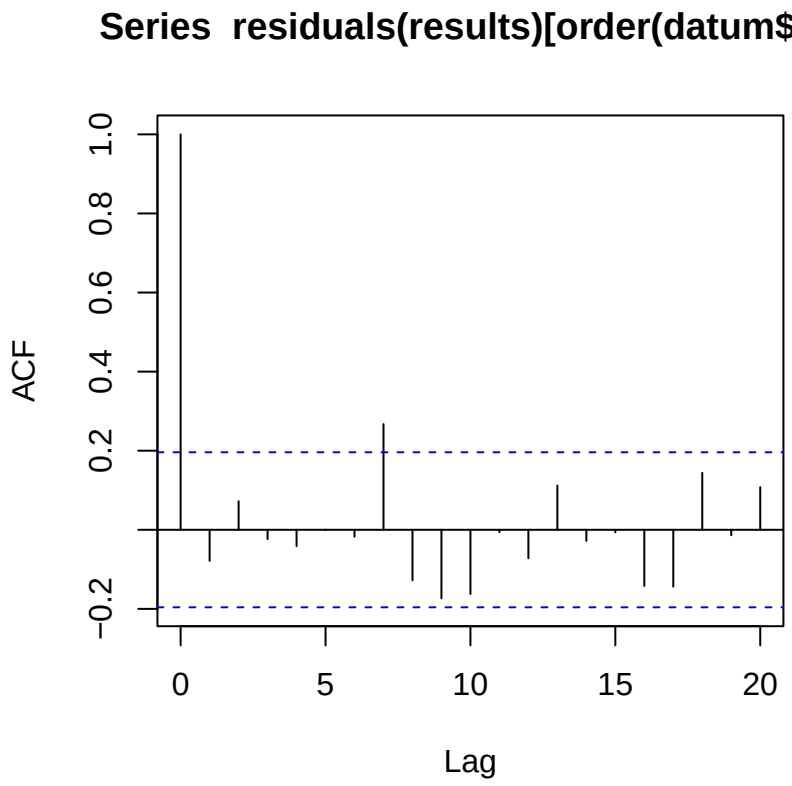
This second graph suggests that the bell curve might be a bit 'skinny' – which is called **kurtosis**. Kurosis is when the shape of your distribution is not-normal. It may have *skew* in one direction (a tail), more observations in the middle (bulge), or be 'skinny' and have longer tails on both sides. This is a rare problem for you to have to deal with, but you can do 'weighted regression' in 'lm()' to account for uneven variance in the model itself.

Autocorrelation functions (ACF)

The **autocorrelation function** measures how similar each datapoint is to the other datapoints that are behind it in the dataframe. It compares residuals at different *lags* in the dataframe.

- At a lag of zero, we expect high correlation, because we compare each point to itself.
- At lags of 1 or more, if there is autocorrelation, then the ACF metric will be high and positive (> 0.2) when comparing each a datapoint to points lagged behind it (i.e., high correlation to nearby data in the dataset).
- If there is no autocorrelation, then the ACF metric will randomly be positive and negative and usually within 0.2 of 0.

```
# Autocorrelation function for normal data
acf(residuals(results)[order(datum$x)])
```



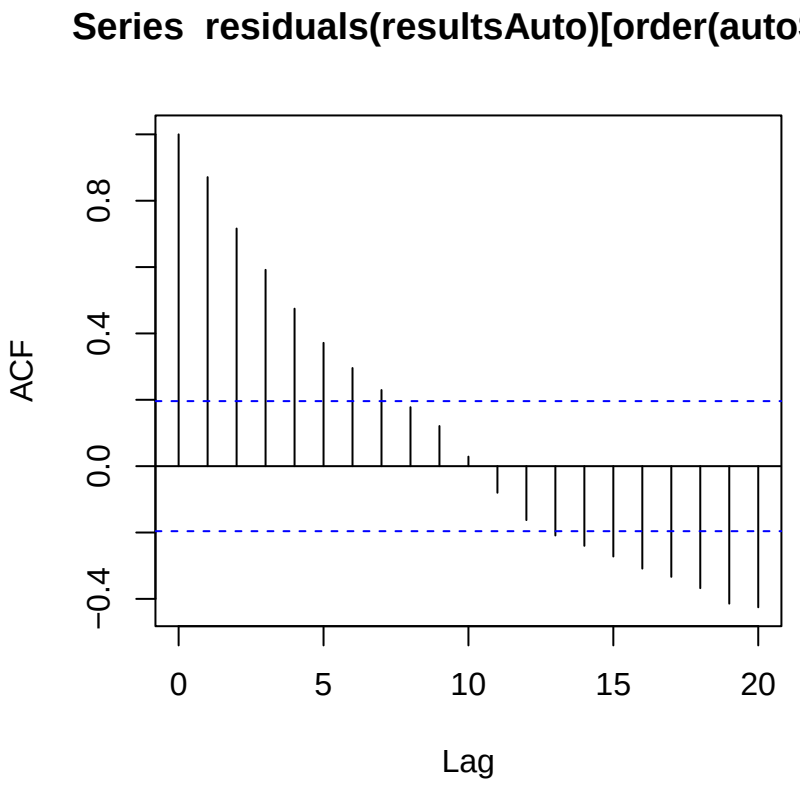
When we run this, we have to make sure our ACF function operates using the order of the X-variable. This might reflect the biological mechanism with which we might expect autocorrelation to happen. For example, if we were measuring how a powerplant causes pollution in a river, we would order our data using the X-variable 'distance from plant'. So we will index the ACF using the 'order' of the X-variable.

Let's examine the ACF graph a bit more. On the X-axis, we have a 'lag' in comparing the residuals in our data.

- At a lag of 0, we are comparing each point to itself. At a lag of 1, we are comparing each point to the point next to it. At a lag of 2, we compare each point to the point two points away. Etc.
- The vertical bar at each lag is the mean estimate of the ACF function. The length of the bar indicates how correlated each value is to it's lagged comparison. This estimates how correlated they are to eachother (i.e., an r^2 value). At a lag of 0, we are comparing each point to itself, so the correlation is 1 (perfect correlation). This is meaningless to us but provides contextual reference.
- The **dotted blue line** is basically a p-value line. If any of your vertical bars cross this blue line, it indicates that you have significant autocorrelation at that lag.
- For example, note the ACF estimate at a lag = 7 was above the line! But, be careful here. If you randomly have autocorrelation at one lag, think about whether this makes biological sense. Instead, we are looking for **obvious and egregious** evidence of autocorrelation.

What does obvious autocorrelation look like...?

```
# Autocorrelation function for normal data
acf(residuals(resultsAuto)[order(auto$x)])
```

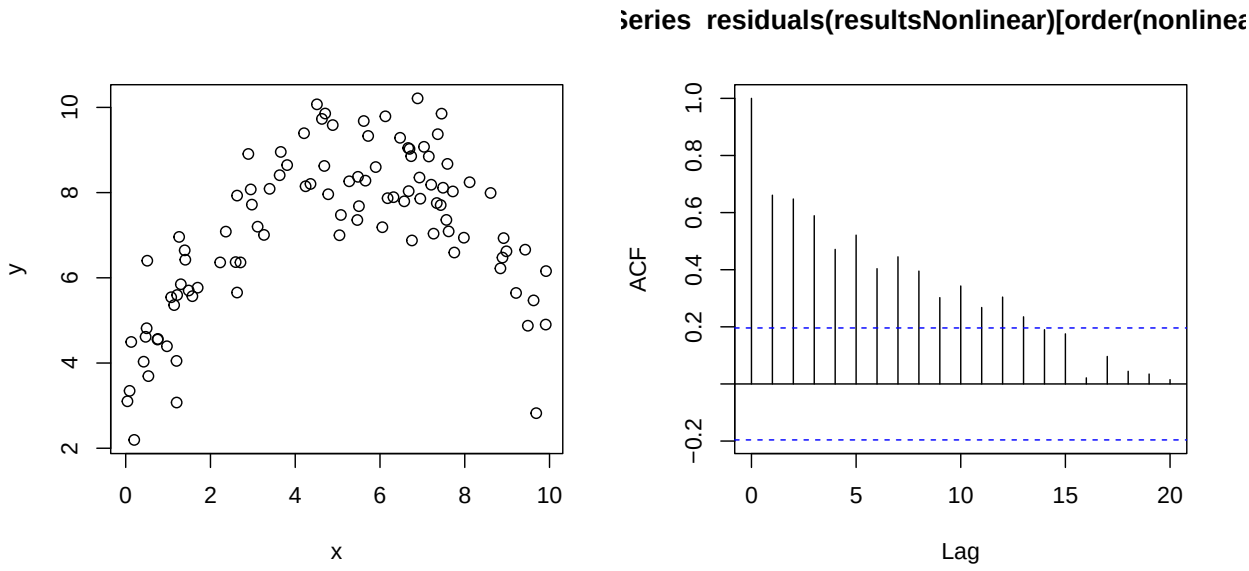


This is obvious autocorrelation. We see that each point is highly correlated with the value next to it! E.g., r^2 values are high: greater than 0.8! It isn't until a lag of ~8 that we see the ACF values dip below the blue line. And then eventually the have negative autocorrelation at a lag of 15 and beyond.

- In ecology, this behavior in data is pretty uncommon. However, one notable example where it might appear involves predator-prey population cycles, like the famous population dynamics between snowshoe hare-lynx in boreal forests of North America.

This chart is best for testing for autocorrelation. But, I mentioned last class, that sometimes these charts can indicate autocorrelation *when data are nonlinear*:

```
# Two graphs
par(mfrow=c(1,2))

# Recall the scatterplot for the nonlinear data
plot(y ~ x, data = nonlinear)

# Autocorrelation function with correct order
acf(residuals(resultsNonlinear)[order(nonlinear$x)])
```



Remember, the autocorrelation function is comparing residuals. Is one residual similar to the residuals next to it?

Since a linear model was fit through these data, we have clusters of negative and positive residuals, and the ACF will pick up on this autocorrelation. However, this observed autocorrelation is not due to true autocorrelation in the system, but rather our use of linear model being fit to nonlinear data. Our line doesn't fit well! And here, the ACF is telling us that we should double-check whether our data are linear.

If a nonlinear line was fit through these data, the residuals would be normal and the ACF would not detect autocorrelation. We may learn how to do this in a few weeks.

Concluding thoughts

I use these graphs to test the assumptions of linear regression model. You do have to run the regression first! It's part of your exploratory process. You should think about whether your data are linear or not before starting an analysis. I may make a 'field guide' for considerations when running an analysis, and/or maybe we will work on that together during the last class.

Simulating data for this lecture

Code to simulate the above datasets is included here:

```
### Code for simulating data to be analyzed in this lecture

# Set the seed for reproducibility
set.seed(123)

## Simulate data with no assumption violations
n <- 100
x1 <- rnorm(n, mean = 20, sd = 10)
y1 <- 5 + 4 * x1 + rnorm(n, mean = 0, sd = 8)

# Create dataframe
```

```r
datum <- data.frame(x = x1, y = y1)

# Save the CSV file
write.csv(datum, "lecture_6_good_data.csv")


## Simulate data with a violation of normality
n <- 100
x <- rnorm(n, mean = 20, sd = 3)
y <- 10 + 25 * x + rnorm(n, mean = 0, sd = 8)^2

# Create dataframe
datum <- data.frame(x = x, y = y)

# Save the CSV file
write.csv(datum, "lecture_6_nonnormal_data.csv")


## Simulate data with a violation of linearity
n <- 100
x <- runif(n, 0, 10)
#x <- sort(x)
y <- 3 + 2 * x - 0.18 * x^2 + rnorm(n, mean = 0, sd = 1)

# Create dataframe
datum <- data.frame(x = x, y = y)

# Save the CSV file
write.csv(datum, "lecture_6_nonlinear_data.csv")


## Simulate data that are heteroscedastic
n <- 100
x1 <- runif(n, 0, 10)
y1 <- 5 + 4 * x1 + rnorm(n, mean = 0, sd = 1 * x1)

# Create dataframe
datum <- data.frame(x = x1, y = y1)

# Save the CSV file
write.csv(datum, "lecture_6_heteroscedastic_data.csv")


## Simulate data that are autocorrelated
n <- 100
x1 <- runif(n, 0, 10)

# Sort x1 from low to high
x1 <- sort(x1)

# Simulate error for each value using the mean of the previous value
error <- matrix(NA, length(x1), 1)
error[1,1] <- rnorm(1, mean = 0, sd = 1)
```

```r
for (i in 2:length(x1)){
  error[i,1] <- rnorm(1, mean = error[i-1, 1], sd = 1)
}

# Create y values
y1 <- 3 + 2 * x1 + error

# Create dataframe
datum <- data.frame(x = x1, error = error, y = y1)

# Save the CSV file
write.csv(datum, "lecture_6_autocorrelated_data.csv")
```

–go to next lecture–