Exploration librairie swirl : Residuals

```
> swirl()

 Welcome to swirl! Please sign in. If you've been here before, use the same  name as you did
then. If you are new, call yourself something unique.

What shall I call you? jlbellier

 Please choose a course, or type 0 to exit swirl.

1: Regression Models
2: Statistical Inference
3: Take me to the swirl course repository!

Selection: 1

 Please choose a lesson, or type 0 to return to course menu.

 1: Introduction
 2: Residuals
 3: Least Squares Estimation
 4: Residual Variation
 5: Introduction to Multivariable Regression
 6: MultiVar Examples
 7: MultiVar Examples2
 8: MultiVar Examples3
 9: Residuals Diagnostics and Variation
10: Variance Inflation Factors
11: Overfitting and Underfitting
12: Binary Outcomes
13: Count Outcomes

Selection: 2

                                                                                          0%
Residuals. (Slides for this and other Data Science courses may be found at github
https://github.com/DataScienceSpecialization/courses. If you care to use them, they must be
downloaded as a zip file and viewed locally. This lesson corresponds to
Regression_Models/01_03_ols. Galton data is from John Verzani's website,
http://wiener.math.csi.cuny.edu/UsingR/)
                      ===                                                                 3%

This lesson will focus on the residuals, the distances between the actual children's heights
and the estimates given by the regression line. Since all lines are characterized by two
```

parameters, a slope and an intercept, we'll use the least squares criteria to provide two equations in two unknowns so we can solve for these parameters, the slope and intercept.

```
                    =======                                  6%
```

The first equation says that the "errors" in our estimates, the residuals, have mean zero. In other words, the residuals are "balanced" among the data points; they're just as likely to be positive as negative. The second equation says that our residuals must be uncorrelated with our predictors, the parents' height. This makes sense - if the residuals and predictors were correlated then you could make a better prediction and reduce the distances (residuals) between the actual outcomes and the predictions.

```
                  =========                                  9%
```

We'll demonstrate these concepts now. First regenerate the regression line and call it fit. Use the R function lm. Recall that by default its first argument is a formula such as "child ~ parent" and its second is the dataset, in this case galton.

```
> fit <- lm(child~parent, data=galton)
```

All that practice is paying off!

```
               ==============                               12%
```

Now we'll examine fit to see its slope and intercept. The residuals we're interested in are stored in the 928-long vector fit$residuals. If you type fit$residuals you'll see a lot of numbers scroll by which isn't very useful; however if you type "summary(fit)" you will see a more concise display of the regression data. Do this now.

```
> summary(fit)

Call:
lm(formula = child ~ parent, data = galton)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.8050 -1.3661  0.0487  1.6339  5.9264 

Coefficients:
            Estimate Std. Error t value Pr(>t)    
(Intercept) 23.94153    2.81088   8.517   <2e-16 ***
parent       0.64629    0.04114  15.711   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.239 on 926 degrees of freedom
Multiple R-squared:  0.2105,	Adjusted R-squared:  0.2096 
F-statistic: 246.8 on 1 and 926 DF,  p-value: < 2.2e-16
```

Perseverance, that's the answer.

```
              ================                              16%
```

First check the mean of fit$residuals to see if it's close to 0.

```
> mean(fit$residuals)
[1] -2.359884e-15
```

All that hard work is paying off!

```
              ====================                                    19%
```

Now check the correlation between the residuals and the predictors. Type "cov(fit$residuals, galton$parent)" to see if it's close to 0.

```
> cov(fit$residuals,galton$parent)
[1] -1.790153e-13
```

Your dedication is inspiring!

```
            ========================                                  22%
```

As shown algebraically in the slides, the equations for the intercept and slope are found by supposing a change is made to the intercept and slope. Squaring out the resulting expressions produces three summations. The first sum is the original term squared, before the slope and intercept were changed. The third sum totals the squared changes themselves. For instance, if we had changed fit’s intercept by adding 2, the third sum would be the total of 928 4’s. The middle sum is guaranteed to be zero precisely when the two equations (the conditions on the residuals) are satisfied.

```
          ===========================                                 25%
```

We'll verify these claims now. We've defined for you two R functions, est and sqe. Both take two inputs, a slope and an intercept. The function est calculates a child's height (y-coordinate) using the line defined by the two parameters, (slope and intercept), and the parents' heights in the Galton data as x-coordinates.

```
        ===============================                               28%
```

Let "mch" represent the mean of the galton childrens' heights and "mph" the mean of the galton parents' heights. Let "ic" and "slope" represent the intercept and slope of the regression line respectively. As shown in the slides and past lessons, the point (mph,mch) lies on the regression line. This means

1: I haven't the slightest idea.
*2: mph = ic + slope*mch*
*3: mch = ic + slope*mph*

Selection: ***3***

Keep working like that and you'll get there!

```
      ==================================                              31%
```

The function sqe calculates the sum of the squared residuals, the differences between the actual children's heights and the estimated heights specified by the line defined by the given parameters (slope and intercept). R provides the function deviance to do exactly this using a fitted model (e.g., fit) as its argument. However, we provide sqe because we'll use it to test regression lines different from fit.

```
    =====================================                             34%
```

We'll see that when we vary or tweak the slope and intercept values of the regression line which are stored in fit$coef, the resulting squared residuals are approximately equal to the sum of two sums of squares - that of the original regression residuals and that of the tweaks themselves. More precisely, up to numerical error,

```
  =============================                                    38%
```

sqe(ols.slope+sl,ols.intercept+ic) == deviance(fit) + sum(est(sl,ic)ˆ2)

```
  ===============================                                  41%
```

Equivalently, sqe(ols.slope+sl,ols.intercept+ic) == sqe(ols.slope, ols.intercept) + sum(est(sl,ic)ˆ2)

```
  ==================================                               44%
```

The left side of the equation represents the squared residuals of a new line, the "tweaked" regression line. The terms "sl" and "ic" represent the variations in the slope and intercept respectively. The right side has two terms. The first represents the squared residuals of the original regression line and the second is the sum of squares of the variations themselves.

```
  ====================================                             47%
```

We'll demonstrate this now. First extract the intercept from fit$coef and put it in a variable called ols.ic . The intercept is the first element in the fit$coef vector, that is fit$coef[1].

```
> ols.ic <- fit$coef[1]
```

All that hard work is paying off!

```
  ======================================                           50%
```

Now extract the slope from fit$coef and put it in the variable ols.slope; the slope is the second element in thefit$coef vector, fit$coef[2].

```
> ols.slope <- fit$coef[2]
```

That's correct!

```
  =========================================                        53%
```

Now we'll show you some R code which generates the left and right sides of this equation. Take a moment to look it over. We've formed two 6-long vectors of variations, one for the slope and one for the intercept. Then we have two "for" loops to generate the two sides of the equation.

```
  ===========================================                      56%
```

Subtract the right side, the vector rhs, from the left, the vector lhs, to see the relationship between them. You should get a vector of very small, almost 0, numbers.

```
> lhs-rhs
[1]  1.264198e-09  2.527486e-09  3.801688e-09 -1.261469e-09 -2.522938e-09 -3.767127e-09
```

You nailed it! Good job!

```
  ==============================================                   59%
```

You could also use the R function all.equal with lhs and rhs as arguments to test for equality. Try it now.

```
> all.equal(lhs,rhs)
[1] TRUE
```

```
You are doing so well!
  =====================================================================     62%
Now we'll show that the variance in the children's heights is the sum of the variance in the OLS estimates and the variance in the OLS residuals. First use the R function var to calculate the variance in the children's heights and store it in the variable varChild.

> varChild <- var(galton$child)

That's correct!
  ===================================================================       66%
Remember that we've calculated the residuals and they're stored in fit$residuals. Use the R function var to calculate the variance in these residuals now and store it in the variable varRes.

> varRes <- var(fit$residuals)

All that hard work is paying off!
======================================================================      69%
Recall that the function "est" calculates the estimates (y-coordinates) of values along the regression line defined by the variables "ols.slope" and "ols.ic". Compute the variance in the estimates and store it in the variable varEst.

> varEst <- var(est(ols.slope, ols.ic))

You are amazing!
 ======================================================================     72%
Now use the function all.equal to compare varChild and the sum of varRes and varEst.

> all.equal(varChild,varRes+varEst)
[1] TRUE

You are amazing!
 ====================================================================       75%
Since variances are sums of squares (and hence always positive), this equation which we've just demonstrated, var(data)=var(estimate)+var(residuals), shows that the variance of the estimate is ALWAYS less than the variance of the data.
=======================================================================     78%
Since var(data)=var(estimate)+var(residuals) and variances are always positive, the variance of residuals

1: is unknown without actual data
2: is greater than the variance of data
3: is less than the variance of data

Selection: 3

That's the answer I was looking for.
```

```
===========================================================================  81%
```

The two properties of the residuals we've emphasized here can be applied to datasets which have multiple predictors. In this lesson we've loaded the dataset attenu which gives data for 23 earthquakes in California. Accelerations are estimated based on two predictors, distance and magnitude.

```
===========================================================================  84%
```

Generate the regression line for this data. Type efit <- lm(accel ~ mag+dist, attenu) at the R prompt.

```
> efit <- lm(accel ~ mag+dist, attenu)
```

Perseverance, that's the answer.

```
=========================================================================     88%
```

Verify the mean of the residuals is 0.

```
> mean(efit$residuals)
[1] -1.785061e-18
```

You got it right!

```
===========================================================================  91%
```

Using the R function cov verify the residuals are uncorrelated with the magnitude predictor, attenu$mag.

```
> cov(efit$residuals,attenu$mag)
[1] 5.338694e-17
```

Keep up the great work!

```
===========================================================================  94%
```

Using the R function cov verify the residuals are uncorrelated with the distance predictor, attenu$dist.

```
> cov(efit$residuals,attenu$dist)
[1] 5.253433e-16
```

Keep working like that and you'll get there!

```
===========================================================================  97%
```

Congrats! You've finished the course on Residuals. We hope it hasn't left a bad taste in your mouth.

```
===========================================================================100%
```