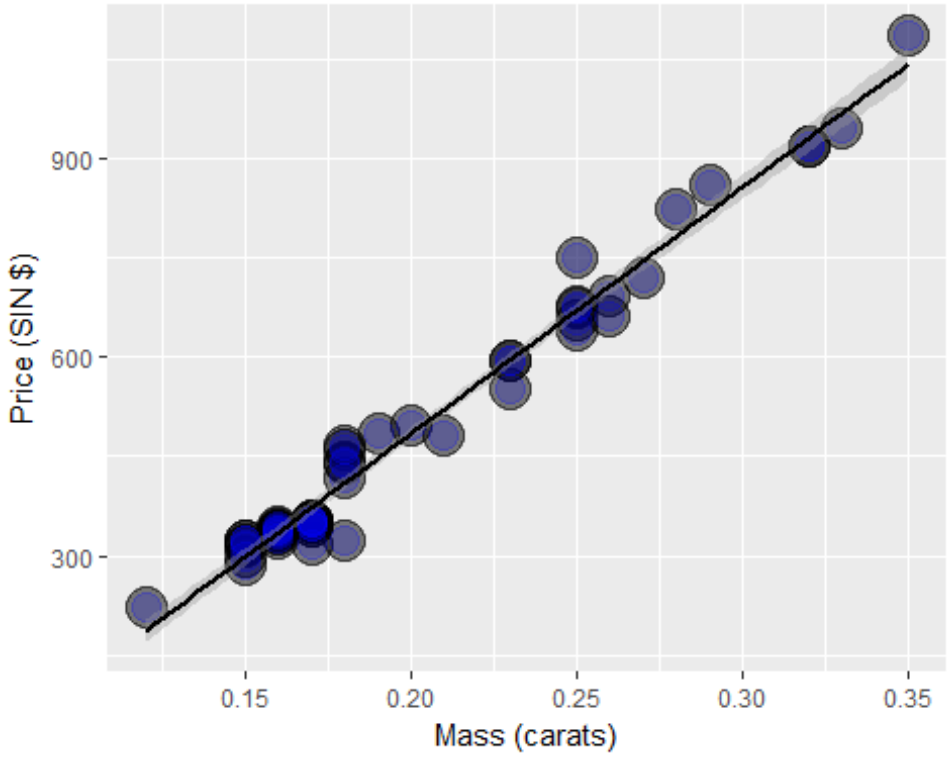
Regression Models (Part VIII: Residuals)

A. Residual variation

Residuals represent variation left unexplained by our model. We emphasize the difference between residuals and errors. The errors unobservable true errors from the known coefficients, while residuals are the observable errors from the estimated coefficients. In a sense, the residuals are estimates of the errors.

Consider again the diamond data set from UsingR. Recall that the data is diamond prices (Singapore dollars) and diamond weight in carats (standard measure of diamond mass, 0.2 g). To get the data use library(UsingR); data(diamond). Recall the data and our linear regression fit looked like the following:

```
library(UsingR)
data(diamond)
library(ggplot2)
g = ggplot(diamond, aes(x = carat, y = price))
g = g + xlab("Mass (carats)")
g = g + ylab("Price (SIN $)")
g = g + geom_point(size = 7, colour = "black", alpha=0.5)
g = g + geom_point(size = 5, colour = "blue", alpha=0.2)
g = g + geom_smooth(method = "lm", colour = "black")
g
```



Recall our linear model was

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

where we are assuming that $\epsilon_i \sim N(0, \sigma^2)$. Our observed outcome is Y_i with associated predictor value, X_i. Let's label the predicted outcome for index i as $\hat{Y}_i$. Recall that we obtain our predictions by plugging our observed X_i into the linear regression equation:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$$

The residual is defined as the difference the between the observed and predicted outcome

$$e_i = Y_i - \hat{Y}_i.$$

The residuals are exactly the vertical distance between the observed data point and the associated point on the regression line. Positive residuals have associated Y values above the fitted line and negative residuals have values below.

Least squares minimizes the sum of the squared residuals, $\sum_{i=1}^{n} e_i^2$. Note that the e_i are observable, while the errors, ϵ_i are not. The residuals can be thought of as estimates of the errors.

B. Properties of the residuals

Let's consider some properties of the residuals. First, under our model, their expected value is 0, $E[e_i] = 0$. If an intercept is included, $\sum_{i=1}^{n} e_i = 0$. Note this tells us that the residuals are not independent. If we know n-1 of them, we know the n^{th}. In fact, we will only have n-p free residuals, where p is the number of coefficients in our regression model, so p=2 for linear regression with an intercept and slope. If a regressor variable, X_i, is included in the model then $\sum_{i=1}^{n} e_i X_i = 0$.

What do we use residuals for? Most importantly, residuals are useful for investigating poor model fit. Residual plots highlight poor model fit.

Another use for residuals is to create covariate adjusted variables. Specifically, residuals can be thought of as the outcome (Y) with the linear association of the predictor (X) removed. So, for example, if you wanted to create a weight variable with the linear effect of height removed, you would fit a linear regression with weight as the outcome and height as the predictor and take the residuals. (Note this only works if the relationship is linear.)

Finally, we should note the different sorts of variation one encounters in regression. There's the total variability in our response, usually called total variation. One then differentiates residual variation (variation after removing the predictor) from systematic variation (variation explained by the regression model). These two kinds of variation add up to the total variation, which we'll see later.

Example

The code below shows how to obtain the residuals.

```
data(diamond)
y <- diamond$price; x <- diamond$carat; n <- length(y)
fit <- lm(y ~ x)
## The easiest way to get the residuals
e <- resid(fit)
## Obtain the residuals manually, get the predicted Ys first
yhat <- predict(fit)
## The residuals are y - yhat. Let's check by comparing this
## with R's build in resid function
max(abs(e -(y - yhat)))
## [1] 9.485746e-13
## Let's do it again hard coding the calculation of Yhat
max(abs(e - (y - coef(fit)[1] - coef(fit)[2] * x)))
## [1] 9.485746e-13
```