Regression Models

Sébastien Plat

Contents

Linear Least Square Regression

Definition

Regression Models are used to understand relathionships / predict outcomes based on existing data.

Let's try to describe the relation between two variables X_i and Y_i as linear, with **Gaussian errors** added to the linear fit (**probabilistic model** for Linear Regression):

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i = \hat{Y}_i + \epsilon_i \quad where\ \epsilon_i\ are\ iid\ N(0, \sigma^2)$$

Line of best fit

We want to find the **line of best fit**, ie. the line that is the best approximation of the given set of data. It is the line that **minimizes the square errors**, ie. the **difference between the predictions and actual outcomes**:

$$Min(\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = Min\ {\epsilon_i}^2 \quad \bullet \quad \beta_1 = Cor(X,Y) \times \frac{Sd(Y)}{Sd(X)} \quad \bullet \quad \beta_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$$

- The line passes through the point $(\bar{X}, \bar{Y})$
- If we choose $\beta_1 = 0$ (reg through the origin), then $\beta_0 = \bar{Y}$
- For centered data $(X_i - \bar{X}, Y_i - \bar{Y})$:
  - the slope is unchanged
  - the regression goes through the origin (as both means equal zero)

Note: For normalized data $\{\frac{X_i - \bar{X}}{Sd(X)}, \frac{Y_i - \bar{Y}}{Sd(Y)}\}$*, the slope is* $Cor(Y, X)$*.*

Intercept

The **intercept** (ITC) β_0 is the **expected value of the response when the predictor is 0**.

It is not always meaningful to the study, so we can use $\tilde{X}_i = X_i - \bar{X}$ instead: the ITC is the expected response at the average X value.

Slope

The **slope** β_1 is the **expected change in response for a 1 unit change in the predictor**.

If we change the unit of X, we must adjust β_1 to keep the same slope: $oldX = \alpha\ newX \implies new\beta_1 = old\beta_1/\alpha$.

Correlation

The **Correlation Coefficient** $Cor(X, Y)$ measures the **strength of the linear relationship** between X and Y. The closer it is from -1 or 1, the stronger the relationship *(0 means no relationship)*.

It is calculated by **normalizing the Covariance** $Cov(X, Y)$, the measure of **how much two random variables change together**, ie. the degree to which two random variables tend to deviate from their expected values in similar ways.

Residuals

Residuals are the difference between the observed outcome Y_i and prediction $\hat{Y}_i$, ie. the part of outcome Y_i **not explained** by its linear association with predictor X. They are estimates of ϵ_i.

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

- $E[e_i] = 0$: the residuals are as likely to be positive as negative
- If an intercept is included, the model goes through $(\bar{X}, \bar{Y})$ so: $\sum_{i=1}^{n} e_i = 0$

Inference and Prediction intervals

We can compute a **Confidence Interval** for β_0 and β_1 estimates under iid Gaussian errors $\epsilon \sim N(0, \sigma^2)$: their variance is a function of σ. From these CI depend:

- intervals for the **regression line** at point x_0
- the **prediction** of what a y would be at point x_0

The standard error for predictions is bigger than for the regression line. It means that **the Confidence Interval for predictions is wider than for the regression line**. Also:

- Both intervals have varying widths
- Least width at $X = \bar{X}$
- We are quite confident in the regression line, so the interval is very narrow
- If we knew β_0 and β_1, this interval would have zero width
- The prediction interval must incorporate the variabilibity of data around the prediciton line
- Even if we knew β_0 and β_1, this interval would still have width

Variability and Correlation

The total variability of the outcome is the **variability around its mean**. It can be split in two:

- **Regression Variability**: variability explained by the Linear Model
- **Error/Residual Variability**: what's leftover around the regression line

$$\sum_{i=1}^{n}(Y_i - \bar{Y})^2 = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2 = \sum_{i=1}^{n} {e_i}^2 + \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2$$

Note: as variances > 0, the variance of residuals is always less than the variance of data

The percentage of the total variability explained by the Linear Model is:

$$R^2 = \frac{Regression\ Variability}{Total\ Variability} = 1 - \frac{Residual\ Variability}{Total\ Variability} = Cor(X, Y)^2$$

Example: Diamonds data

Introduction

The library 'UsingR' includes a set of data called 'diamonds'. It gives the weight and price of 48 diamonds, in carats and Singapore dollars respectively.

Price as F(carat)

We can see how the price of diamonds relates to their weight by a linear model:

```
#calculate estimates
data(diamond)
fit <- lm(price ~ carat, data = diamond)
diamEst <- diamond %>% mutate (price = predict(fit)) # predictions of Y = f(X)
diamRes <- diamond %>% mutate (price = resid(fit)) # residuals of Y = f(X)

# ITC and slope for 1/10th carat increase
fit <- lm(price ~ I(carat*10 - mean(carat*10)), data = diamond)
sumCoef <- summary(fit)$coefficients

qt975 <- c(-1, 1) * qt(.975, df = fit$df)
mCoef <- rbind(c(coef(fit)[1], sumCoef[1,1] + qt975 * sumCoef[1, 2]), # beta0
               c(coef(fit)[2], sumCoef[2,1] + qt975 * sumCoef[2, 2])) # beta1

rownames (mCoef) <- c("beta0", "beta1")
colnames (mCoef) <- c("fit", "lwr", "upr")

round(mCoef,2)
```

```
         fit    lwr    upr
beta0 500.08 490.83 509.33
beta1 372.10 355.64 388.57
```

- **$500.08** is the expected price for the average sized diamond (0.20 carats)
- **$372.10** is the expected change in price for every 1/10th of a carat increase in mass
- With 95% confidence, we estimate that a 0.1 carat increase results in a **$355.60 to $388.60 increase** in price

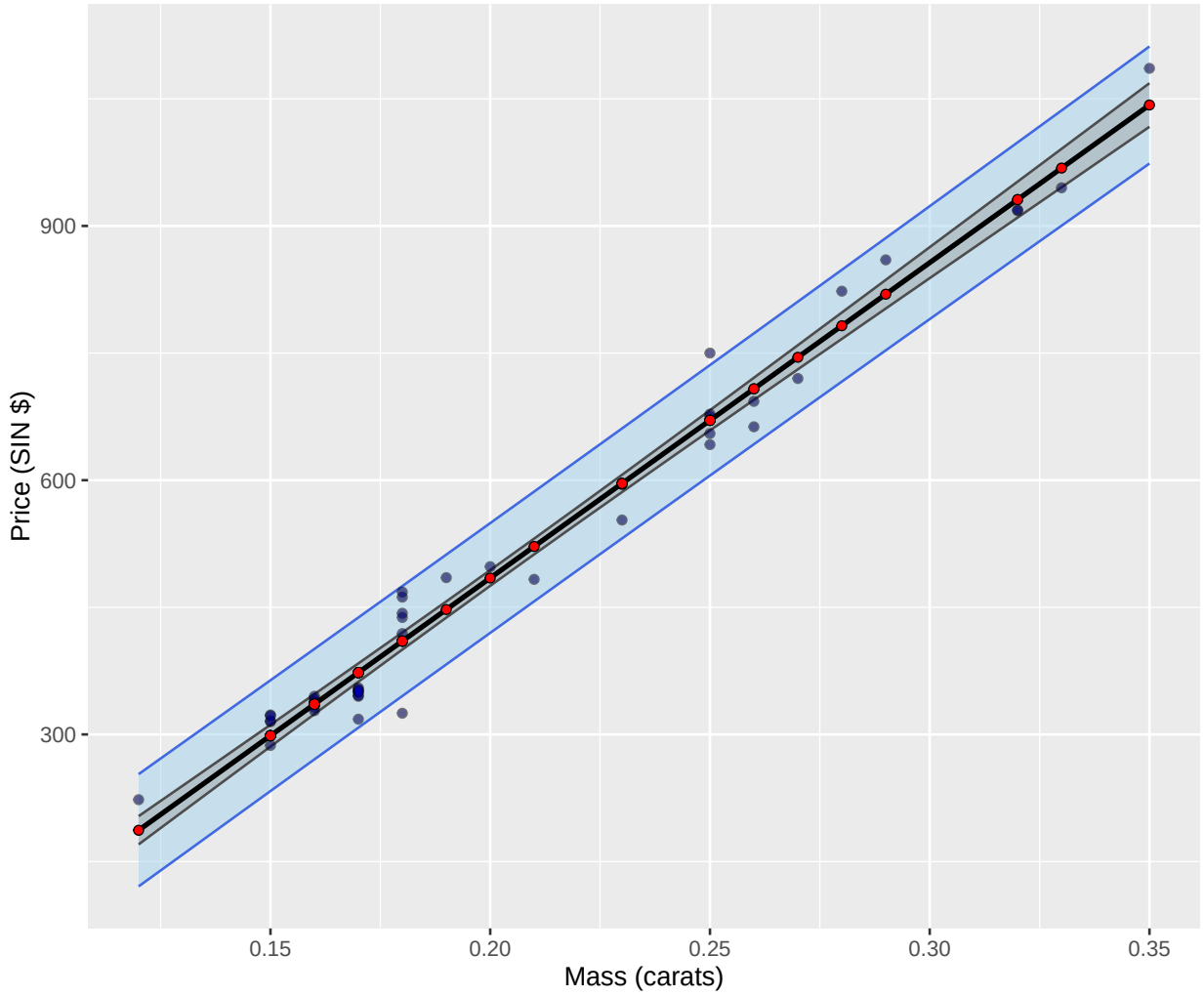
```
#calculate correlation coefficient
summary(fit)$r.squared
```

```
[1] 0.9782608
```

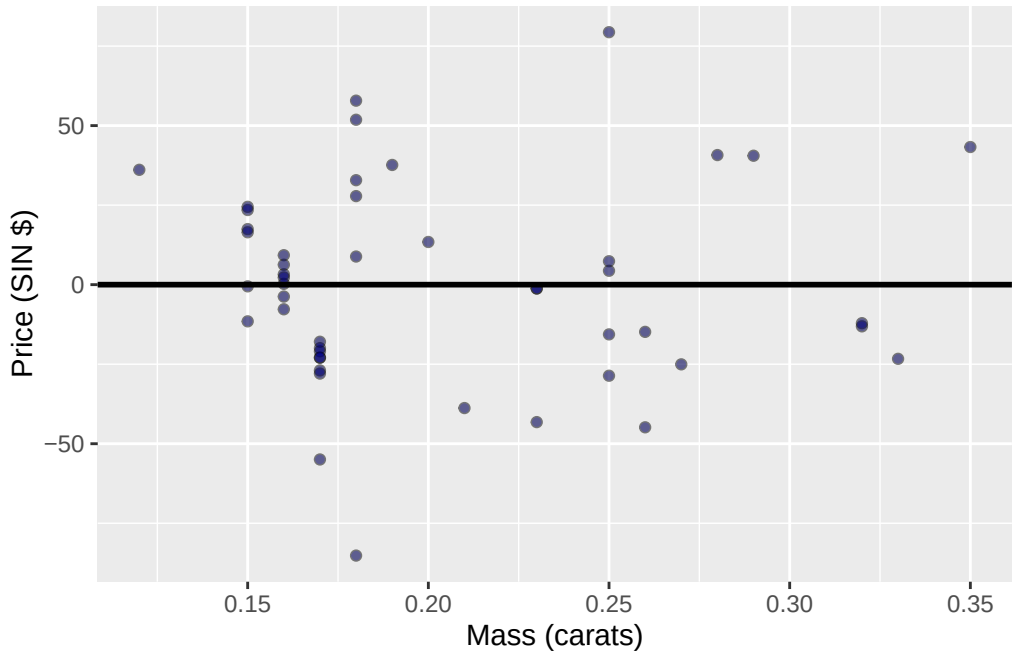
Approx. 97.8% of the variance can be explained by our model.

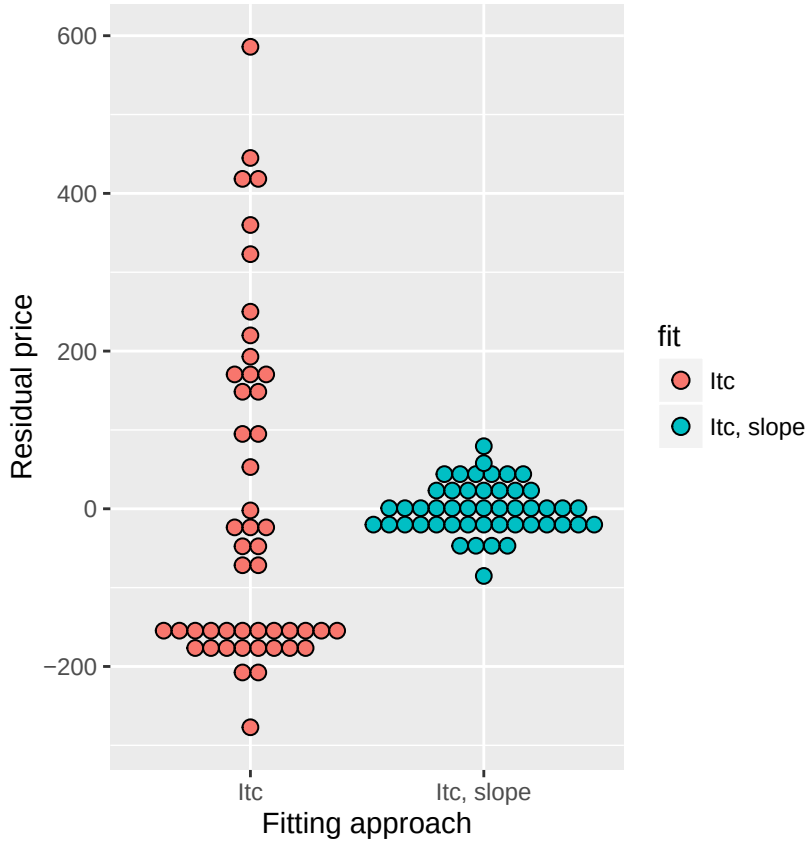
Linear regression model

```
#creating a vector of x values, from Xmin to Xmax
xVals <- seq(min(diamond$carat), max(diamond$carat), by = .01)
newdata <- data.frame(carat = xVals)
fit <- lm(price ~ carat, data = diamond)

#computing predictions for the regression line ("confidence")
p1 <- cbind(newdata, predict(fit, newdata, interval = ("confidence"))) # x, fit, lwr, uwr

#computing predictions for price estimations ("predictions")
p2 <- cbind(newdata, predict(fit, newdata, interval = ("prediction"))) # x, fit, lwr, uwr
```



Residuals


50
0
−50
Price (SIN $)
0.15
0.20
0.25
0.30
0.35
Mass (carats)


600
400
200
0
−200
Residual price
ltc
ltc, slope
Fitting approach
fit
ltc
ltc, slope

Assumptions Diagnostics

Introduction

The estimation and inference from the regression model depend on several assumptions that need to be checked.

- **errors**: assumed to be iid $\sim N(0, \sigma)$, ie. independant, with same variance and normally distributed
- **model**: assumed to be a vector $\hat{Y} = X\beta$
- **outliers**: observations that do not fit the model

Note: for more information: + SO question (link) + Examples of regression plots (link)

Definitions

Terms used for diagnostics:

- **Fitted values** $\hat{Y}_i$: model prediction for each data point
- **Residuals** e_i: difference between the fitted values & the actual outcome for each data point
- **Standardized residuals**: residuals rescaled (mean=0, sd=1); accounts for individual residual variances

```
predict(fit)   # fitted values
resid(fit)     # residuals
rstandard(fit) # standardized residuals
rstudent(fit)  # studentized residuals - will not be detailed
```

R plot.lm

The four plots can be used to diagnose specific problems:

- The upper left plot shows whether the **wrong model was fitted** (e.g., a line versus a parabola)
- The upper right plot shows whether the residuals are **normally distributed**
- The lower left plot shows whether the data are **homoskedastic**
- The lower right plot shows whether there are **influential outlier**

Errors constant variance / Uncorrelation to fit

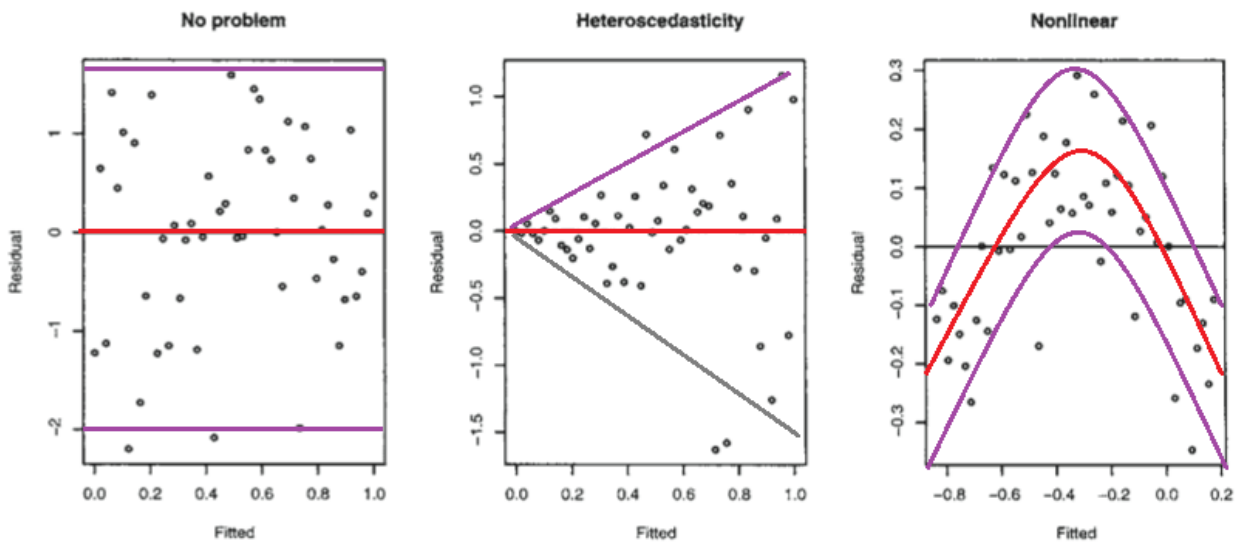
The assumption of **homoskedasticity** (constant errors variance) is required to make the OLS estimator the **optimal solution**, ie. the minimum variance unbiased estimator.

It is not possible to check the assumption of constant variance just by examining the residuals alone - some will be large and some will be small, but this proves nothing. We need to check whether the variance in the residuals is related to some other quantity.

Residuals vs Fitted values

The simplest diagnostic plot displays **residuals** e_i **versus fitted values** $\hat{Y}_i$.

```
plot (lm, which = 1)
```



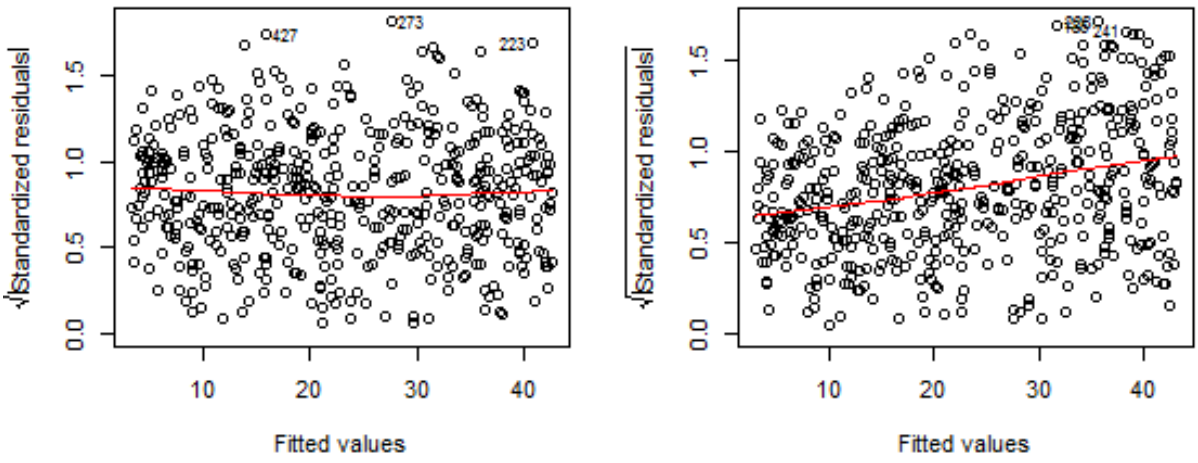
- plot 1 shows correct assumptions: errors are symetrically distributed around mean 0
- plot 2 shows **heteroskedasticity**: errors variance is not constant but increases with X
- plot 3 shows **nonlinearity**: errors mean is not constant, so linear model is not a good fit

Note: Plot 3 indicates that the fitted line doesn't describe how Y behaves as X changes.

Scale-Location plot

Another way to assess homoskedasticity is a **scale-location plot**. In case of constant variance, the line will be **flat, not sloped**. Otherwise, a data transformation (e.g., log transformation) should be tried.

```
plot (lm, which = 2)
```

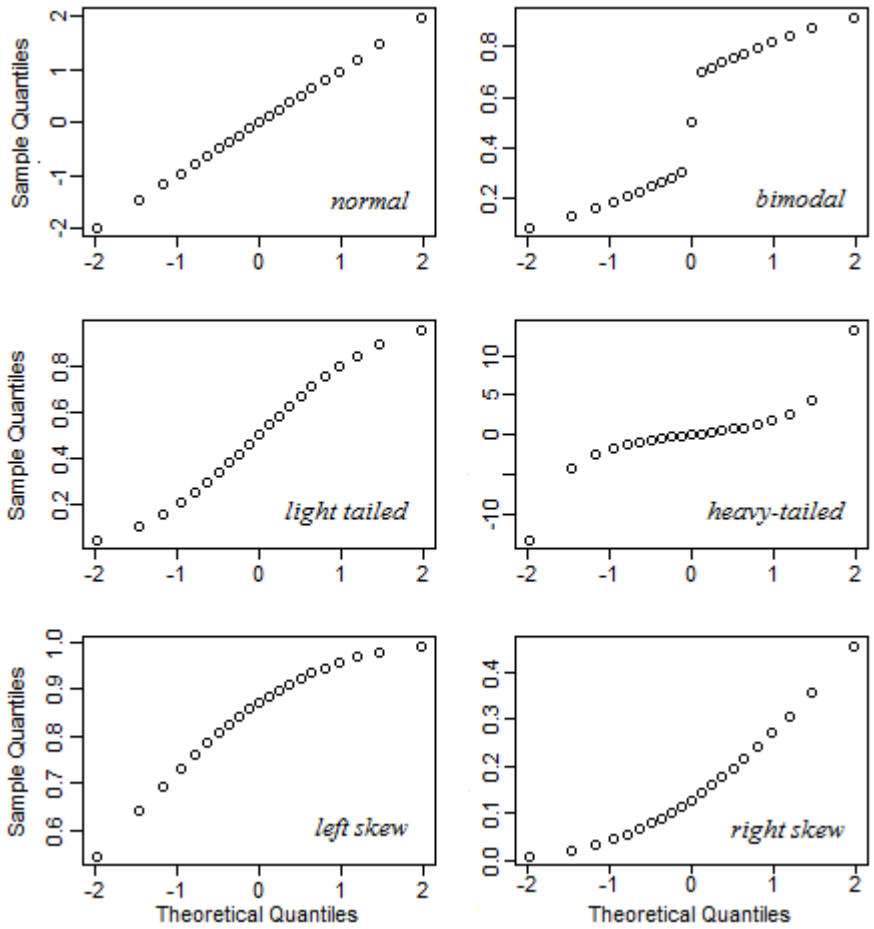


Normality - QQ Plot

The tests and confidence intervals are based on the assumption of **normal errors**. It can be checked by using a **Q-Q plot**, which compares the **residuals to "ideal" normal observations**.

When the errors are not normal, least squares estimates may not be optimal. They will still be best linear unbiased estimates, but other robust estimators may be more effective. Mild nonnormality can safely be ignored and the larger the sample size the less troublesome the nonnormality.

```
plot (lm, which = 3)
```



- For light-tailed distributions, the consequences of nonnormality can be ignored.
- For long-tailed errors, we might just accept the nonnormality and:
  - base the inference on the assumption of another distribution
  - use resampling methods such as the bootstrap or permutation tests
  - use robust methods, which give less weight to outlying observations
- For skewed errors, a transformation of the response may solve the problem

Outliers - Cook's distance

An **outlier** is a data point **far from the mean of X and/or Y**. It is said to:

- have **leverage** when far from **mean of X**
- be **influential** when it **significantly affect the fit**

Outliers may or may not belong in the data. They may represent real events or they may be spurious. In any case, they should be examined.

The change in coefficients induced by including/excluding a sample is a simple measure of its influence.

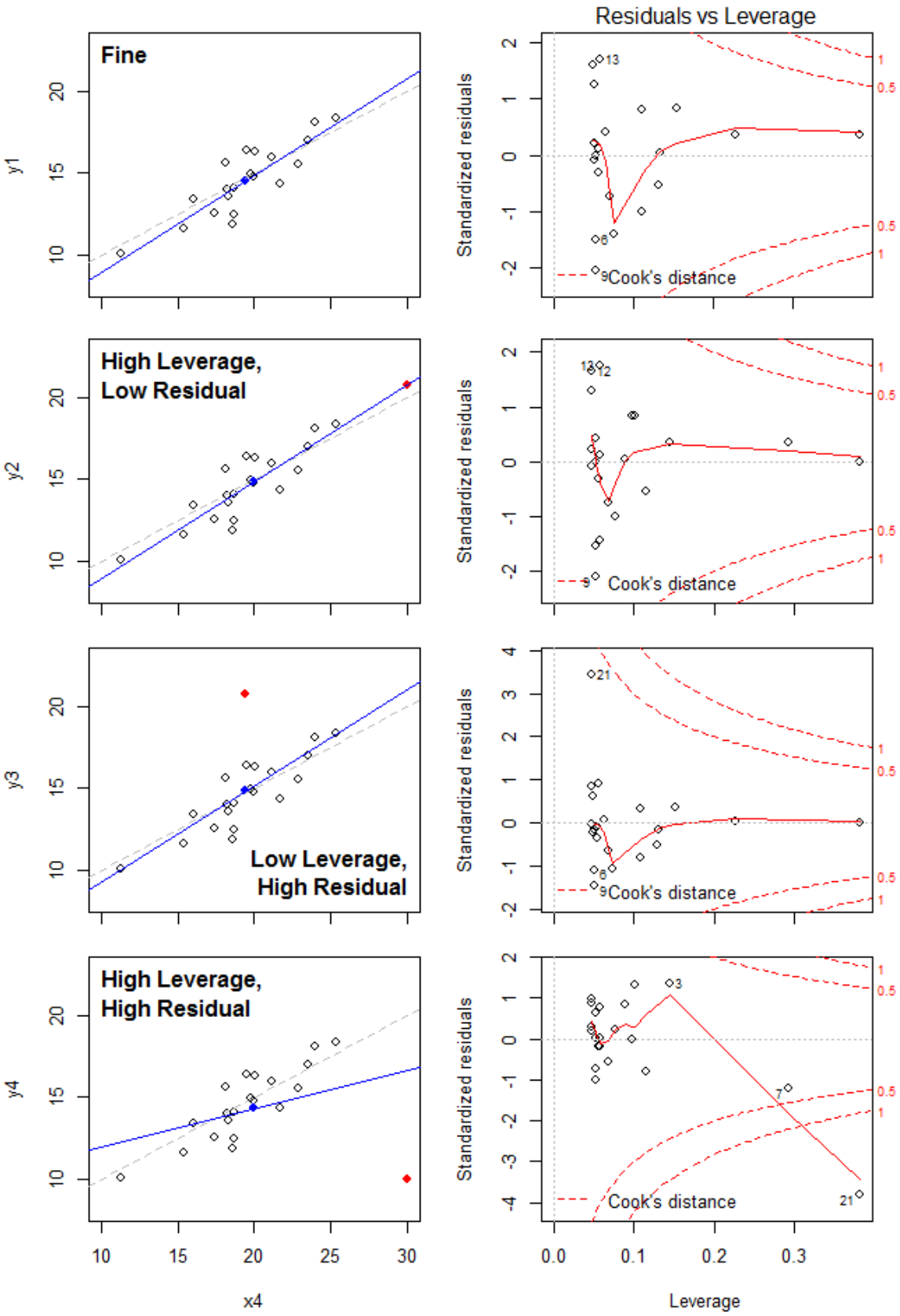
The difference in predicted values between fits with/without a data point is called Cook's distance. It can be used to easily identify highly influencial outliers.

```
cooks.distance(lm)
plot (lm, which = 5)
```

The plots on the left show:

- the data
- the center of the data $(\bar{X}, \bar{Y})$ with a blue dot
- the underlying data generating process with a dashed gray line
- the model fit with a blue line
- the special point with a red dot

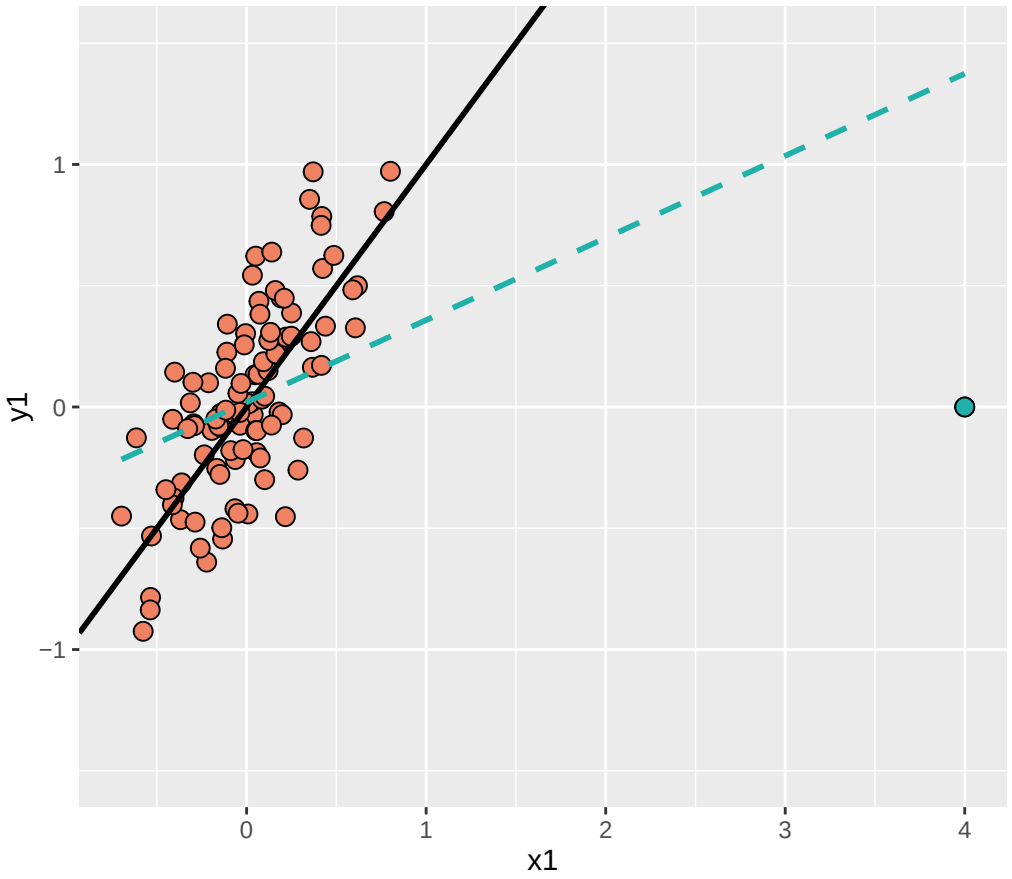
On the right are the corresponding residual-leverage plots; the special point is 21. The model is badly distorted primarily in the fourth case where there is a point with high leverage and a large (negative) standardized residual.


Fine
High Leverage,
Low Residual
Low Leverage,
High Residual
High Leverage,
High Residual
y1
y2
y3
y4
x4
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage

Example: single outlier

Let's look at a random set of points where a clear outlier c(4, 0) has created an artificial but strong regression relationship where there shouldn't be one.

```
set.seed(1000)
n <- 100
x0 <- rnorm(n, sd = .3); y0 <- x0 + rnorm(n, sd = .3)
x1 <- c(4, x0); y1 <- c(0, y0)
fit <- lm (y1 ~ x1)
```

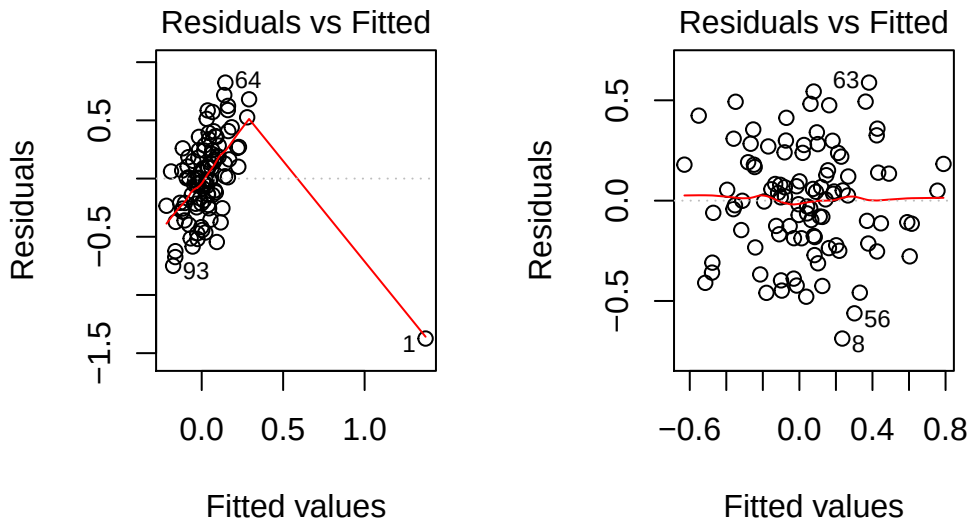


Residuals versus fitted values

The simplest diagnostic plot displays residuals versus fitted values. Residuals should be:

- uncorrelated with the fit
- independent
- (almost) identically disributed with mean zero

```
fit <- lm (y1 ~ x1)
y2 <- y1[-1]; x2 <- x1[-1]
fitno <- lm (y2 ~ x2)
```

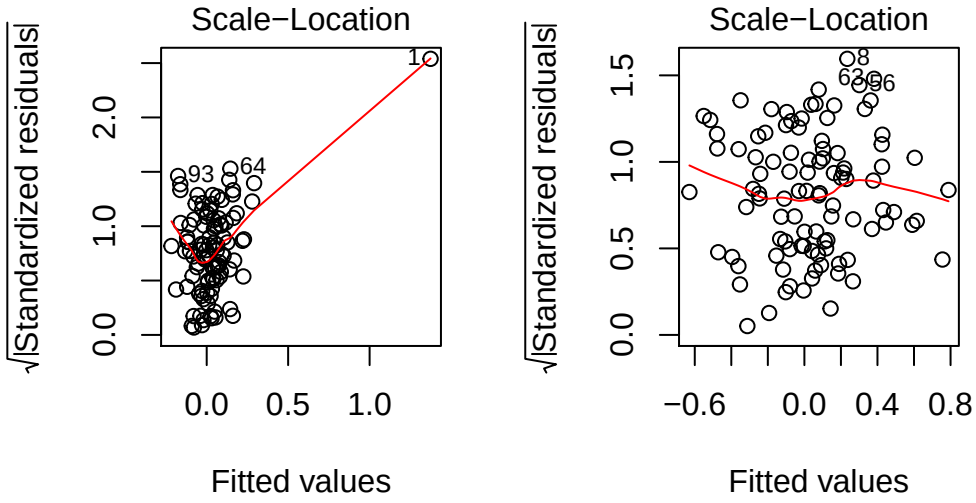


On the left, there is a linear pattern involving all but one residual and the fit. The Residuals vs Fitted plot labels certain points with their row names or numbers, numbers in our case. The influential outlier is row N°1.

Without the outlier, on the right, the plot has none of the patterned appearance: residuals are independently and (almost) identically distributed with zero mean, and are uncorrelated with the fit.

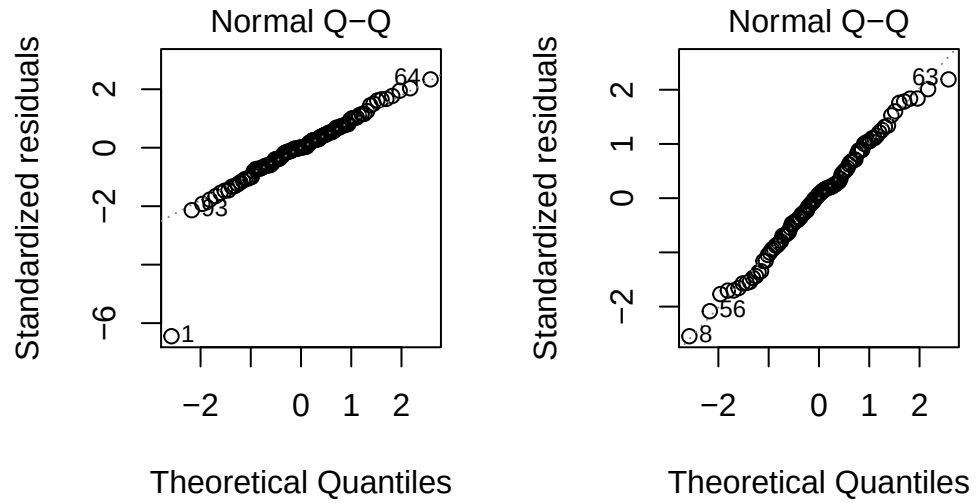
Scale-Location plot

A Scale-Location plot shows the square root of standardized residuals against fitted values.



QQPlot

The assumption that residuals are approximately normal is implicit in linear models. Since standardized residuals adjust for individual residual variances, a QQ plot of standardized residuals against normal with constant variance is of interest. On the left, the outlier is about -7 standard deviations from the mean.



Cook's distance

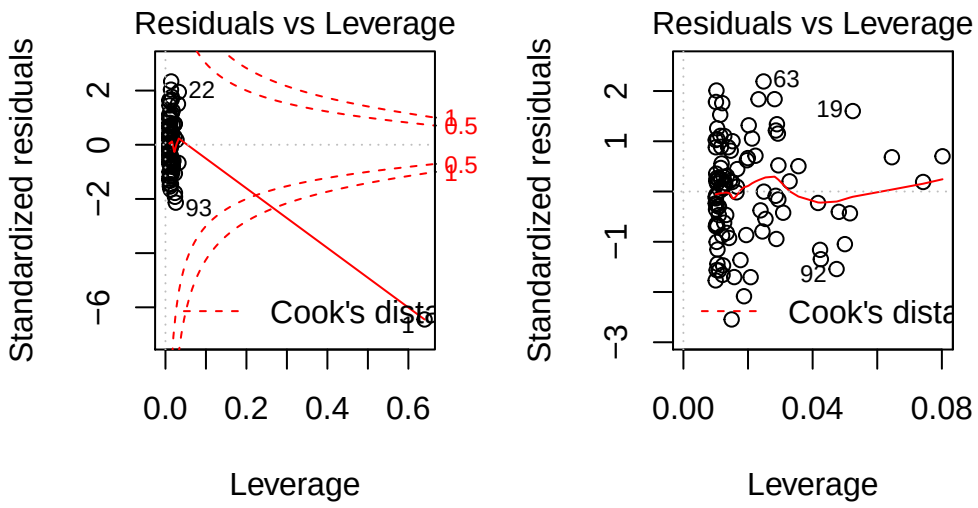
Cook's distance is essentially the sum of squared differences between values fitted with and without a particular sample.

It is normalized (divided by) residual sample variance times the number of predictors ,which is 2 in our case (the intercept and x.).

It essentially tells how much a given sample changes a model.

```
dy <- predict(fit) -
      predict(fitno, newdata=data.frame(x2=x1))

cbind(sum(dy^2)/(2*summary(fit)$sigma^2), # Cook's distance
      cooks.distance(fit)[1])
```

```
     [,1]    [,2]
1 36.9623 36.9623
```



Multivariable Regression Analysis

Introduction

The general linear model extends simple linear regression (SLR) by adding terms linearly into the model. Using a vector representation (usually $X_{1i} = 1$, so that an intercept is included):

$$X_i{}^T = \begin{bmatrix} X_{1i} = 1 \\ \vdots \\ X_{pi} \end{bmatrix} \quad and \quad \beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} \quad then \quad E[Y_i] = \mu_i = X_i\beta$$

$$Y_i = \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_p X_{pi} + \epsilon_i = \sum_{k=1}^{p} X_{ik}\beta_j + \epsilon_i = X_i\beta + \epsilon_i \quad where\ \epsilon_i \sim N(0, \sigma^2)$$

Least squares minimize $\sum_{i=1}^{n} (Y_i - X_i\beta)^2$

Estimates

Multivariate regression estimates are exactly those of a SLR through the origin, having removed the linear relationship of the other variables from both the regressor and response. In this sense, multivariate regression "adjusts" a coefficient for the linear impact of the other variables.

- Fitted responses: $\hat{Y}_i = X_i\hat{\beta}$
- Residuals: $e_i = Y_i - \hat{Y}_i$
- Variance estimate: $\hat{\sigma}^2 = \frac{1}{n-p}\sum_{i=1}^{n} e_i^2$
- Coefficients have standard errors, $\hat{\sigma}_{\hat{\beta}_k}$, and $\frac{\hat{\beta}_k - \beta_k}{\hat{\sigma}_{\hat{\beta}_k}}$ follows a T distribution with $n-p$ degrees of freedom
- Predicted responses have standard errors and we can calculate predicted and expected response intervals

Interpretation of the coefficient

The interpretation of a multivariate regression coefficient is the **expected change in the response per unit change in the regressor**, holding **all of the other regressors fixed**.

Binary variables

We can apply linear models to compare k+1 groups: $Y_i = \beta_0 + X_{i,1}\beta_1 + \ldots + X_{i,k}\beta_k + \epsilon_i$

where each $X_{i,j}$ is binary

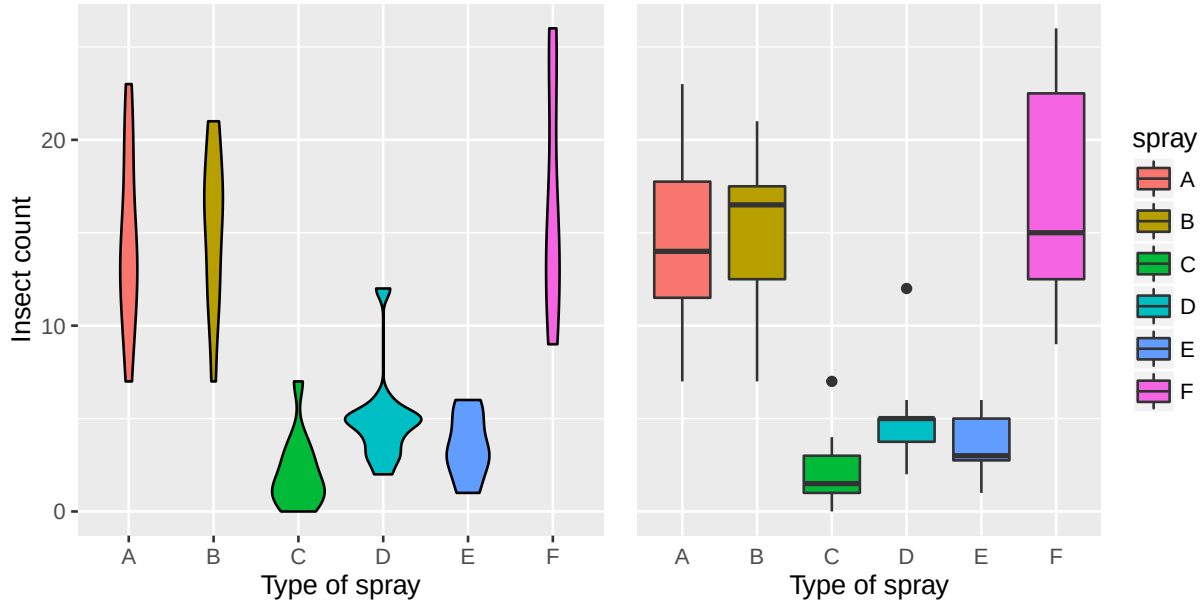
- 1 if measurement X_i is in group j
- 0 otherwise

It means that:

- for measurements in group 0, $E[Y_i] = \beta_0$
- for all other measurements, $E[Y_i] = \beta_0 + \beta_j$

So β_j is interpreted as the increase or decrease in the mean comparing group j to group 0, called the **reference group**.

Example: Insect Sprays data




```
# means of each group
summary(lm(count ~ spray - 1, data = InsectSprays))$coef
```

```
        Estimate Std. Error   t value     Pr(>|t|)
sprayA 14.500000   1.132156 12.807428 1.470512e-19
sprayB 15.333333   1.132156 13.543487 1.001994e-20
sprayC  2.083333   1.132156  1.840148 7.024334e-02
sprayD  4.916667   1.132156  4.342749 4.953047e-05
sprayE  3.500000   1.132156  3.091448 2.916794e-03
sprayF 16.666667   1.132156 14.721181 1.573471e-22
```

```
# variations in means compared to group C (dafault: vs group A)
spray2 <- relevel(InsectSprays$spray, "C")
summary(lm(count ~ spray2, data = InsectSprays))$coef
```

```
             Estimate Std. Error  t value     Pr(>|t|)
(Intercept)  2.083333   1.132156 1.840148 7.024334e-02
spray2A     12.416667   1.601110 7.755038 7.266893e-11
spray2B     13.250000   1.601110 8.275511 8.509776e-12
spray2D      2.833333   1.601110 1.769606 8.141205e-02
spray2E      1.416667   1.601110 0.884803 3.794750e-01
spray2F     14.583333   1.601110 9.108266 2.794343e-13
```

Notes:

- Counts are bounded from below by 0, violates the assumption of normality of the errors
- Variance does not appear to be constant
- Poisson GLM are more adapted for fitting count data

Example: Swiss data

We want to compare the relationship between Agriculture and Fertility in Catholic vs Protestant provinces. We can do it by using binary variables with an interaction term:

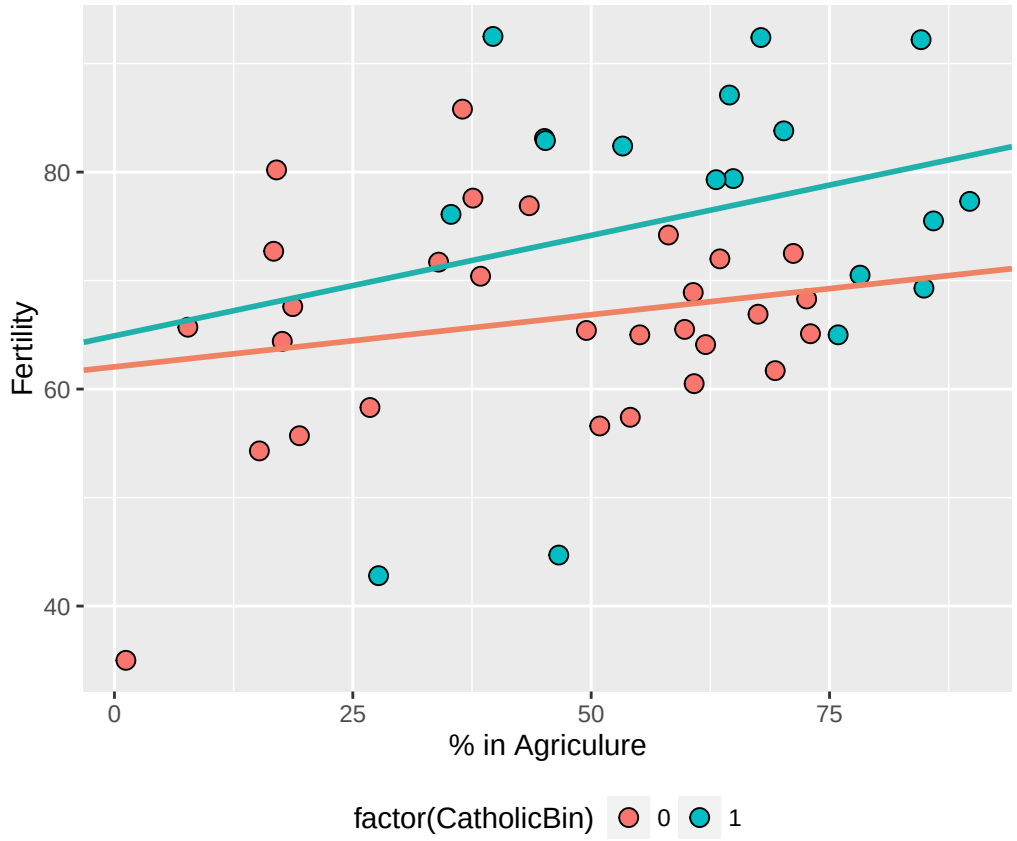
$$E[Y|X_1, X_2] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2$$

where:

- X_1 is the Agriculture percentage
- X_2 is the Catholic binary variable

This model gives us two ITC and two slopes.

```
data(swiss)

# Provinces are either majority Catholic or majority Protestant
# hist(swiss$Catholic)
# So we can create a group to identify them clearly
swiss <- mutate (swiss, CatholicBin = 1 * (Catholic > 50))

# Then we apply our model with an interaction term
fit <- lm (Fertility ~ Agriculture * factor (CatholicBin), data=swiss)
```



Model adjustments

Model selection

Our interest lies in parsimonious, interpretable representations of data that enhance our understanding of the phenomena under study. Please note that different models may have different purposes: prediction VS studying mechanisms VS trying to establish causal effects, etc.

- (Known knowns) Regressors that we know we should include, and have
- (Known Unknowns) Regressors that we would like to include, but don't have
- (Unknown Unknowns) Regressors that we don't even know about that we should have included

Omitted-Variable Bias

Omitted-Variable Bias (OVB) occurs when a model **incorrectly leaves out one or more important factors**. The "bias" is created when the model compensates for the missing factor by over- or underestimating the effect of the others (source: Wikipedia).

- the omitted variable must be relevant for the outcome prediction
- the omitted variable must be correlated with another variable included in the model (ie. $Cov(z, x) \neq 0$)
- in that case, the error term is correlated with the regressors, which creates the bias in the OLS estimator

Note: a variable that correlates with both the dependent variable and the independent variable is called a ***confounding variable****.*

Randomizing the samples can help to uncorrelate the outcome with confounding variables.

Including unnecessary variables

Including a new variable to a model:

- mechanically increases R^2, even if not relevant to the outcome
- mechanically decreases the SSE, even if not relevant to the outcome

When the new variable is correlated to other regressors, the actual SE of these increases significantly.

This phenomenon is called **variance inflation**.

Variance inflation factors (VIF)

The Standard Error of regressors coefficients depends on σ, the Standard Deviation of the residuals. We do not know σ, so we can only estimate the increase in the actual SE for including a new regressor. However, σ drops out when calculating SE ratios. It means that we can check the variance inflation when sequentially adding new variables.

The **variance inflation factor** (VIF) is the increase in the variance for the ith regressor compared to the ideal setting where it is orthogonal (ie. uncorrelated) to the other regressors.

Uncorrelating regressors

The problems of bias due to excluded regressors and variance inflation both involve correlated regressors. Methods exist to convert regressors to an equivalent uncorrelated set, such as **factor analysis** or **principal componenent analysis**. The downside is that using converted regressors may make **interpretation difficult**.

ANOVA for Nested model testing

Linear regression minimizes the squared difference between predicted and actual observations, ie. **minimizes the variance of the residuals**. If an additional predictor **significantly reduces the residual's variance**, it is deemed **important** (p-value of an F-test: significant or not).

ANOVA stands for **ANalysis Of Variance**. In the case of nested model testing, it checks if adding new regressors improves the model by checking the ratio of variances. ANOVA is sensitive to the assumption that **residuals are approximately normal**. If they are not, we could get a small p-value for that reason. The **Shapiro-Wilk test** can check is this assumption holds: normality is its null hypothesis.

*Note: the F-statistic is the same as the t-statistic for a t-test, but to check if a **group** of variables are jointly significant.*

Example: Swiss data

```
library(car)
fit <- lm(Fertility ~ . , data = swiss)
vif(fit) # variance inflation factors
```

```
     Agriculture      Examination        Education         Catholic
        2.287787         3.972467         2.898462        13.405099
Infant.Mortality      CatholicBin
        1.107804        14.575205
```

```
fit1 <- lm(Fertility ~ Agriculture, data = swiss)
fit3 <- update(fit, Fertility ~ Agriculture + Examination + Education)
fit5 <- update(fit, Fertility ~ Agriculture + Examination + Education + Catholic + Infant.Mortality)
anova(fit1, fit3, fit5) # anova for nested model testing
```

```
Analysis of Variance Table

Model 1: Fertility ~ Agriculture
Model 2: Fertility ~ Agriculture + Examination + Education
Model 3: Fertility ~ Agriculture + Examination + Education + Catholic +
    Infant.Mortality
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1     45 6283.1
2     43 3180.9  2    3102.2 30.211 8.638e-09 ***
3     41 2105.0  2    1075.9 10.477 0.0002111 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
c(shapiro.test(fit3$residuals)$p.value,
  shapiro.test(fit5$residuals)$p.value) # check for normality residuals
```

```
[1] 0.3359947 0.9317582
```

For both fit3 and fit5, the Shapiro-Wilk test fails to reject the null hypothesis of normality, so the conclusions of the F-test hold: both models improve the fit of the Swiss data.

Generalized Linear Models

Introduction

The GLM generalizes linear regression for cases where the response variables are not normally distributed:

- the linear model is related to the response variable via a link function
- the magnitude of the variance of each measurement is a function of its predicted value

A few common GLM:

- **Bernoulli**: models that predict a probability of making a yes/no choice (log(odds) varies linearly)
- **Poisson**: responses always positive and varying over a wide range (log(response) varies linearly)

As an example, we can try to predict beach attendance as a function of temperature.

- the probability of beach attendance:
  - would typically be modeled with a **Bernoulli distribution** with a log-odds or **logit link function**
  - would predict a constant rate of increased odds in beach attendance
  - an increase of 10 degrees leads to a doubling in beach attendance odds (from 2:1 to 4:1, etc.)
  - a decrease of 10 degrees leads to an halfing in beach attendance odds (from 4:1 to 2:1, etc.)
- the predicted number of beach attendees:
  - would typically be modeled with a **Poisson distribution** and a **log link**
  - would predict a constant rate of increased beach attendance
  - an increase of 10 degrees leads to a doubling in beach attendance
  - a decrease of 10 degrees leads to an halfing in beach attendance

(source: Wikipedia)

Note: **odds** express the **ratio of events to non-events** in the long run. For example, 4:1 means that the probability of an event occurring is 4 times the probability that is does not.

Logistic regression

Binary GLMs come from trying to model outcomes that can take only two values.

For $Y_i \sim Bernoulli(\mu_i)$, the linear model η_i is applied to the logit of the outcome: $\eta = log(\frac{\mu}{1-\mu}) = log(odds)$. So the probability that an event occurs is:

$$\mu_i = \frac{exp(\eta_i)}{1 + exp(\eta_i)}$$

Poisson regression

Many data take the form of unbounded count data (number of calls to a call center, number of cars crossing a bridge, Percent of hits to a website from a country, etc). Poisson distribution can also be used to model binomials with small probability of success p and large number of samples n.

For $Y_i \sim Poisson(\mu_i)$, the linear model η_i is applied to the log of the outcome: $\eta = log(\mu)$. So the count of an event is:

$$\mu_i = exp(\eta_i)$$

Note: the count can be expressed as $\mu_i = \lambda_i t$, where λ_i is the occurence rate.

The assumptions for a Poisson distribution is that $\text{mean}(Y_i) = \text{var}(Y_i) = \mu_i$, so plotting residuals vs fitted values should show an upward trend.

ANOVA for Nested model testing

Deviance extends the idea of checking for residuals variance reduction to generalized linear regression, using (negative) log likelihoods in place of variance. It means we can use ANOVA for GLM Nested models testing:

```
anova(glm1, glm2, glm3)
```

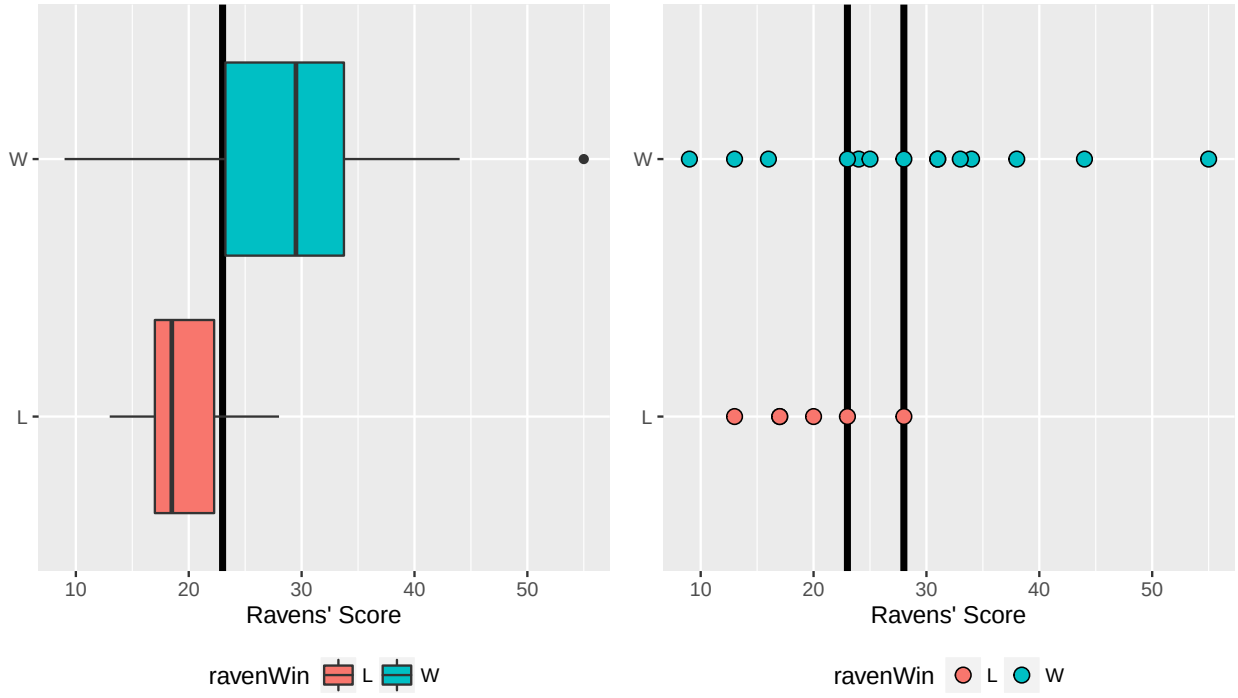
The returned value is a quantile of a chi-square distribution with df(glm1) - df(glm2) degrees of freedom, under the null hypothesis that the additional predictors bring no value to the model (ie. that their coefficients are zero).

To confidently reject this hypothesis, we would want the returned value to be larger than the 95th percentile of chi-square distribution with df(glm1) - df(glm2) degree of freedom:

```
qchisq(0.95, df_diff)
```

Binomial example: Raven Wins

Introduction



We can see that the Ravens tend to win more when they score more points:

- 3/4 of their losses occur when they score less than 23 points
- 3/4 of their wins occurs when they score more than 23 points

We also see a fairly rapid transition in the Ravens' win/loss record between 23 and 28 points:

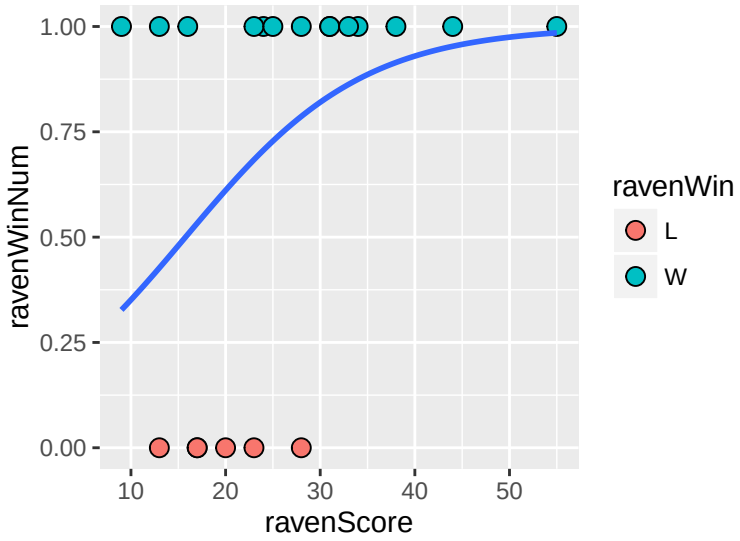
- for 23 points and less they win about half their games (4 out of 9)
- between 24 and 28 points they win 3 of 4
- for above 28 points they win them all

From this, we get a very crude idea of the correspondence between points scored and the probability of a win: an S shaped curve.

GLM

Let's apply a GLM where log odds of a win depend linearly on the score: $log(p/(1-p)) = b0 + b1 * score$. We can do it easily in R:

```
# returns b0 and b1 which maximize the likelihood of our observations
mdl <- glm(ravenWinNum ~ ravenScore, family="binomial", data=ravensData)
```



Let's have a closer look at the model predictions for low scores:

```
# predict returns the log odds for scores 0, 3, 6 and 9
lodds <- predict(mdl, data.frame(ravenScore=c(0, 3, 6, 9)))

# we can convert them into probabilities
prob <- exp(lodds)/(1+exp(lodds)); names (prob) <- c(0,3,6,9)
round(prob,2)
```

```
   0    3    6    9
0.16 0.20 0.26 0.33
```

The model estimates that the Ravens have 16% chance to win a game scoring 0, or 16:84 / 4:21 odds, which is clearly erroneous. It performs poorly for low scores due to lack of data: minimum score was 9 and we only have results for 20 games.

Coefficients

Let's look more closely at its coefficients:

```
mcoef <- summary(mdl)$coefficients
mcest <- c(exp(mcoef[1,1]), exp(mcoef[2,1])); names(mcest) <- c("b0","b1")
round(mcest,2)
```

```
  b0   b1
0.19 1.11
```

The coefficients estimate log odds as a linear function of points scored. It means that:

- $\exp(b0)$ is the odds of winning with a score of 0: $16/84 \simeq 0.19$
- $\exp(b1)$ increases the odds of winning for each point scored: 1.11, ie. 11%

However, the coefficients have relatively large standard errors, which lead to large 95% confidence intervals:

```
round (exp(confint(mdl)), 3)
```

```
            2.5 % 97.5 %
(Intercept) 0.006  3.106
ravenScore  0.996  1.303
```

The lower confidence bound on the odds of winning with a score of 0 is near zero, which seems much more realistic than the 16/84 figure of the maximum likelihood model. On the other hand, the lower bound of exp(b1) is smaller than 1, which suggests that the odds of winning would decrease slightly with every additional point scored. This is obviously unrealistic.

ANOVA

In fact, the GLM version of analysis of variance will show that if we ignore scores altogether, we don't do much worse.

Let's compare the deviance of ravenScore from the null glm to the 95th percentile of chi-square distribution with one degree of freedom (19 - 18):

```
comp <- c(anova(mdl)$Deviance[2], qchisq(0.95, 1)); names(comp) <- c("Deviance","Chi-Square")
round(comp,3)
```

```
  Deviance Chi-Square
     3.540      3.841
```

3.540 is close to but less than the 95th percentile threshold 3.841, so ravenScore adds very little to a model which just guesses that the Ravens win with probability 70% (their actual record that season), ie. odds 7 to 3:

```
mdl0 <- glm(ravenWinNum ~ 1, binomial, ravensData)
m0coef <- summary(mdl0)$coefficients # only the intercept
m0odds <- exp(m0coef[1]) # odds of winning
m0odds/(1+m0odds) # proba of winning
```

```
[1] 0.7
```

Poisson example: daily visits to website

Introduction

Poisson distribution is well suited to website visits, as they tend to occur independently, one at a time, at a certain average rate. We will define λ, the **expected rate of occurrence**, as the **average number of hits per day**. Of course, as the web site becomes more popular, lambda will grow. In other words, **lambda will depend on time**.

We will use **Poisson regression** to model this dependence:

- daily visits to a website as its popularity grows
- the percent of visits coming from another site

In a Poisson regression, the $log(\lambda)$ is assumed to be a linear function of the predictors. As we try to model the growth of visits to a website, we will use time as a predictor: $log(\lambda) = b0 + b1 * date$. This implies that:

- the **average number of hits per day** is **exponential with time**: $\lambda = exp(b0) + exp(b1)^date$
- exp(b1) would represent the **percentage of daily growth**

```
str(gaData)
```

```
'data.frame':   731 obs. of  4 variables:
 $ date       : Date, format: "2011-01-01" "2011-01-02" ...
 $ visits     : num  0 0 0 0 0 0 0 0 0 0 ...
 $ simplystats: num  0 0 0 0 0 0 0 0 0 0 ...
 $ julian     : atomic  14975 14976 14977 14978 14979 ...
  ..- attr(*, "origin")= Date, format: "1970-01-01"
```

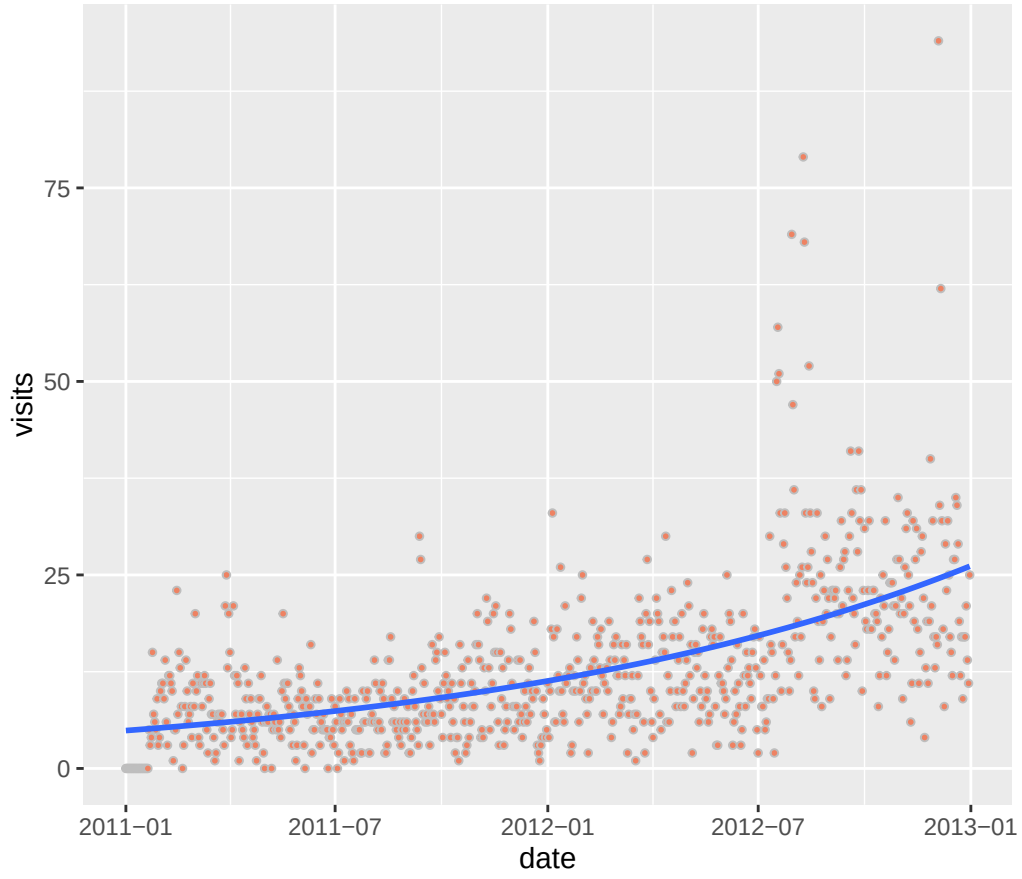
The data has four columns:

- the date & date as julian (easier for manipulation)
- the number of visits from another website, the Simply Statistics blog
- the total number of visits for the day

GLM

Let's apply a GLM where the log of daily visits depends linearly on the date: $log(dailyvisits) = log(\lambda) = b0 + b1 * date$.

```
# returns b0 and b1 which maximize the likelihood of our observations
mdl <- glm(visits ~ date, family="poisson", data=gaData)
```



The figure suggests that our Poisson regression fits the data very well. The blue line is the estimated lambda, or mean number of visits per day. It increased from around 5 in early 2011 to around 10 by 2012, to around 20 by late 2013: approximately doubling every year.

Coefficients

```
summary(mdl)$coefficients
```

```
                Estimate   Std. Error  z value Pr(>|z|)
(Intercept) -32.749835924 8.129718e-01 -40.2841        0
date          0.002293051 5.265741e-05  43.5466        0
```

```
dev <- rbind(c(summary(mdl)$null.deviance,summary(mdl)$df.null),
             c(summary(mdl)$deviance,summary(mdl)$df.residual))
colnames(dev) <- c("deviance", "df"); rownames(dev) <- c("null", "residual")
dev
```

```
         deviance  df
null     5150.018 730
residual 3121.645 729
```

We see that:

- Both coefficients are significant, being far more than two standard errors from zero
- The Residual deviance is also very significantly less than the Null, indicating a strong effect

The Intercept coefficient b0 represents log average hits on R's Date 0, namely January 1, 1970. We will focus more on the date coefficient, since exp(b1) estimates the percentage of increase in daily visits.

```
# 95% confidence interval for exp(b1)
growth <- rbind(exp(confint(mdl, 'date')), # daily growth
                exp(confint(mdl, 'date'))^365) # annual growth
rownames(growth)<- c("daily","yearly")
growth
```

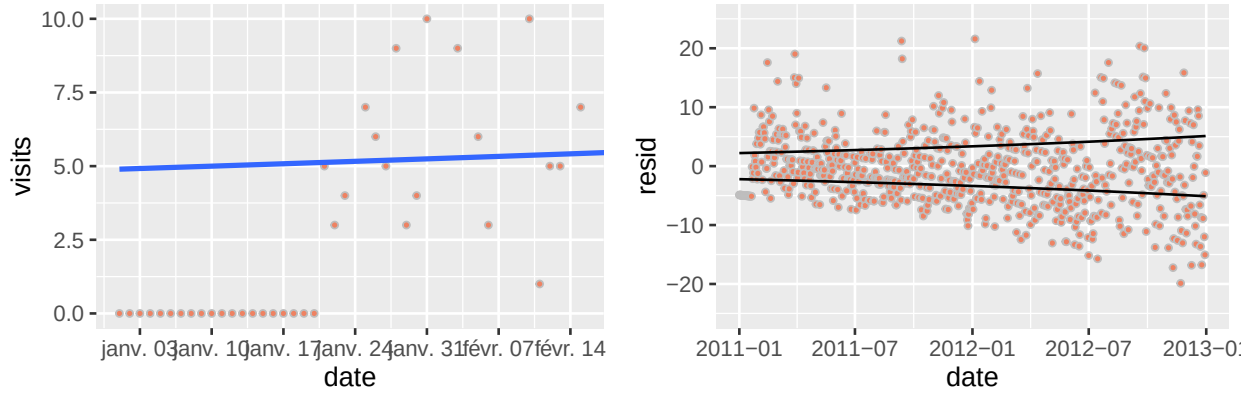
```
          2.5 %   97.5 %
daily  1.002192 1.002399
yearly 2.224130 2.398176
```

Visits are estimated to increase by a factor of between 1.002192 and 1.002399 per day. That is, between 0.2192% and 0.2399% per day. This represents more than a doubling every year.

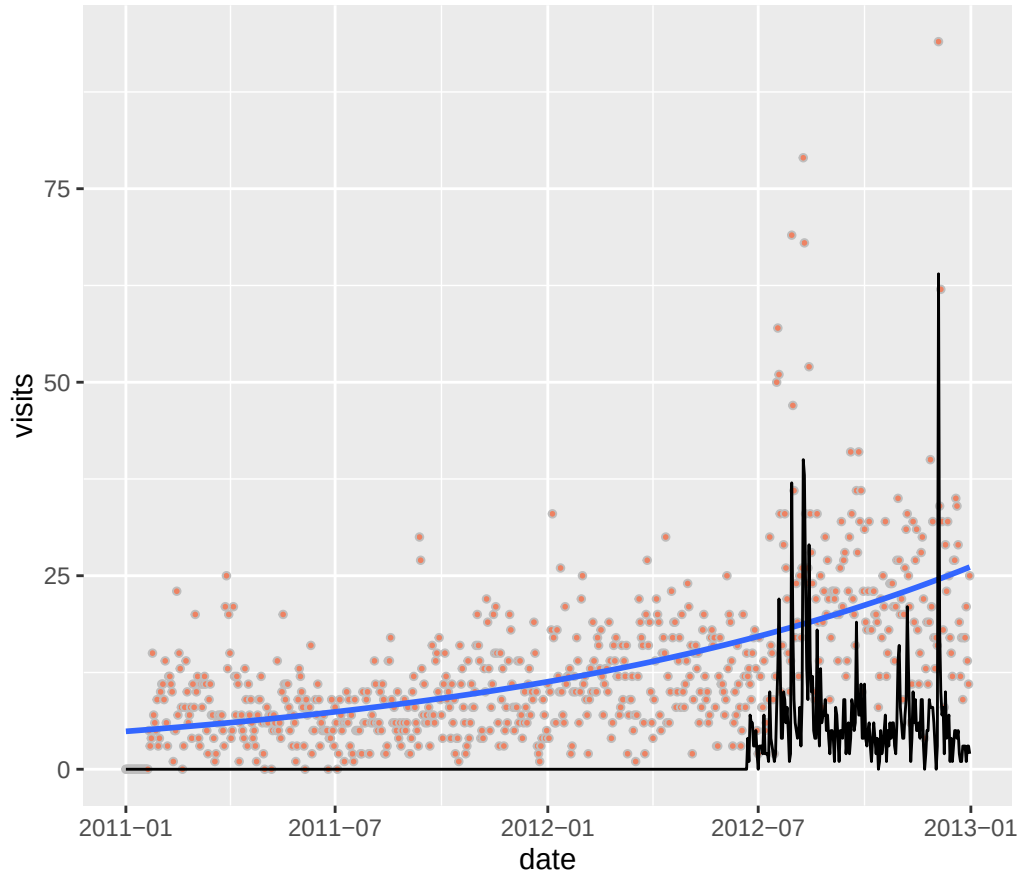
Limitations

Our model looks like a pretty good description of the data, but has some limitations:

- the model doesn't account for bursts of popularity
- 'zero inflation': overestimation for the first two weeks, before the site had any visits
- the data sd may be increasing with lambda faster than a Poisson model allows



During bursts of popularity, the number of visits far exceeds two standard deviations over average. Some are due to mentions on another site, Simply Statistics.



Bursts in popularity

In the figure, the maximum number of visits occurred in late 2012. Visits from the Simply Statistics blog were also at their maximum that day. Let's have a closer look:

```
maxDay <- which.max(gaData[,'visits']) # day with maximum visits
gaData[maxDay,] # details of max day
```

```
          date visits simplystats julian   resid
704 2012-12-04     94          64  15678 69.4522
```

The maximum number of visits, 94, occurred on December 4, 2012:

- 64 came from the Simply Statistics blog (can be considered a special event)
- 30 came from other sources

Let's check if these 30 visits can be considered as normal traffic as estimated by our model.

```
# average number of visits predicted by the model for maxDay
lambda <- mdl$fitted.values[maxDay]

# 95th percentile of this distribution
qpois(.95, lambda)
```

```
[1] 33
```

According to our model, we would see 33 or fewer visits 95% of the time: 30 visits would not be rare. It seems that on December 4, 2012, the very high number of visits was **only due to references from Simply Statistics**.

Visits from another website

A Poisson process can model a proportion by including the denominator of the fraction, or more precisely its log, as an offset.

We may wish to model the proportion of traffic explained by references from Simply Statistics, to evaluate their importance. To avoid division by zero, we'll use simplystats/(visits+1): $log(\lambda) = log(visits+1)+b0+b1*date$ will predict the log of mean visits from Simply Statistics as a proportion of total visits.

```
mdl2 <- glm(simplystats ~ date, family="poisson", offset=log(visits+1), data=gaData)
summary(mdl2)$coefficients
```

```
                 Estimate   Std. Error   z value      Pr(>|z|)
(Intercept) -1.245018e+02 4.5576574374 -27.31707 2.659917e-164
date         7.880592e-03 0.0002920636  26.98246 2.374597e-160
```

Although the estimated coefficients are significantly different from zero, the model is actually not impressive. We can illustrate why by looking at December 4, 2012 once again. On that day, there were 64 actual visits from Simply Statistics, but thatwould be extremely unlikely according to our model:

```
qpois(.95, mdl2$fitted.values[maxDay])
```

```
[1] 47
```